Visualizing Information for Enterprise Architecture Design Decisions using Elastic Stack

Janik Schüssler[1] and Denis Karbstein[2] and Dennis Klein[3] and Alfred Zimmermann[4]

[1] Reutlingen University Germany, Janik.Schuessler@protonmail.com
[2] Reutlingen University Germany, Denis.Karbstein@gmx.de
[3] Reutlingen University Germany, Dennis.Klein87@gmail.com
[4] Reutlingen University Germany, alfred.zimmermann@reutlingen-university.de

Abstract. Digital Enterprise Architecture allows multiple viewpoints on a company's IT landscape. To gain valuable information out of huge amounts of operational data, it is indispensable to have both an understanding of the Operations Architecture and an engine capable of managing Big Data. The mechanism of understanding huge amounts of data is based on three main steps: collect, process and use. The main idea is focused on extracting valuable information out of Big Data to make better design decisions. The Elastic Stack is an open-source solution to comfortably and quickly handle Big Data scenarios.

Keywords: Big Data Analytics, Elastic Stack, Elasticsearch, Logstash, Beats, Kibana, Digital Enterprise Architecture, Operation Architecture, Operational Data, Log Files

1 Introduction

The IT systems of every company generate huge amounts of data every day. This data can be generated at various points (e.g. servers, firewalls, antivirus programs) and companies might not even know which information is contained in the data and how valuable it might be. Therefore, analyzing this data by putting it in context and extracting crucial information out of it is not a typical task within company's IT departments. Most companies do not know how valuable these data are and that they can already be processed with cost-effective analysis tools.

In addition to the missed opportunities with unused data, there are also missed opportunities in Enterprise Architectures. The Enterprise Architecture describes the interaction between processes and IT. There are different dependencies and relations in the IT landscape. Due to lack of analysis of the complex infrastructure, important correlations can be overlooked. Our approach is to show how to use automatically generated data to get conclusions about the Enterprise Architecture and visualize them.

Since modern services are often provided in a highly customized form, it becomes necessary to break down services to more granular Microservices. This can cause illustrations of the holistic Enterprise Architectures to become quite complex. Without properly verified input in descriptively visualized form, this can lead to a very challenging design process.

The goal of this paper is to suggest an approach to those challenges by evaluating Elastic Stack with Enterprise Architecture related use cases and scenarios. The research is focused around the research question '*How can the Elastic Stack process information in order to generate additional value out of information?*'

The research approach started off with theoretical research of sources around the topics **Big Data Analytics** as well as **Enterprise Architecture Design**, while the relevant findings were tested in a practical experiment, which uses an Elastic Stack [6] instance applied to a SWAN server [7].

2 Enterprise Architecture

2.1 Digital Enterprise Composition Architecture

Enterprise Architecture Management (EAM) [4] defines today with frameworks, standards [8], [10], tools and practical expertise a large set of different views and perspectives. We argue in this paper that a new refocused Digital Enterprise Architecture approach should better enable the digitization of products and services. DEA – Digital Enterprise Architecture Reference Cube (Figure 1) is our current extended architectural reference model [9] to support architecture management, engineering and analytics considering a set of multi-perspective viewpoints for Enterprise Architectures. We have introduced new architectural domains for the aspects of data, knowledge, and platform-based ecosystems. The architectural domain mainly concerned in this paper is the extended operation architecture with aspects of operation analytics and visualization.

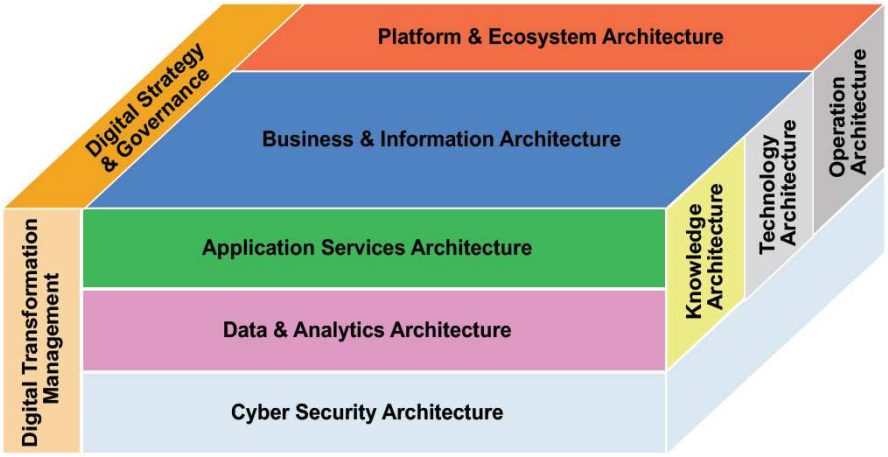

Figure 1: Digital Enterprise Architecture Reference Cube

Digital Enterprise Architecture should be both holistic and easily adaptable to support micro-granular structures like IoT [9] and the digital transformation with new business models and technologies, like social software, Big Data, services computing with cloud computing, mobility platforms and systems, security systems and semantics support.

DEA is more specific than existing architectural standards of EAM – Enterprise Architecture Management [8] and [10] extend these architectural standards for Digital Enterprise Architectures with services and cloud computing. DEA provides a holistic classification model with ten integral architectural domains. These architectural domains cover specific architectural viewpoint descriptions [8] in accordance to the orthogonal dimensions of both architectural layers and architectural aspects [10]. The Open Group Architecture Framework [10] provides the basic blueprint and structure for our extended service-oriented Digital Enterprise Architecture domains.

2.2 Operations Architecture

Digital Enterprise Architectures describes an approach to the organization, development and use of distributed capabilities for digital transformation. Both economic and technological strategies and decisions have an impact on the enterprise architecture as well as on individual elements and their relationship to each other. The management of the IT Architecture is implemented in the **Operations Architecture**. It unites tools and technology used to maintain, monitor and support the architecture, but also data artifacts and flows which contain crucial indications about the current status of a system in terms of health, security, resilience, etc.

In the context of this research, the term **operational data** includes all types of information produced by an Enterprise Architecture instance at runtime. A few examples of operational data are: log files (e.g.: web server access logs, java virtual machine logs, application log files), object data (e.g.: database content, object instances) or infrastructure metrics (e.g.: RAM utilization, network utilization).

A first assumption is that each piece of operational information can be allocated to a certain component in the Enterprise Architecture design. Allocating data to its possible origin might require manual work, since the component which creates the data isn't necessarily the component, the data refers to. For example, an Apache2 web server (Application Component) writes access logs which then contain information about a user request (Business Service)

Second, it is assumed, that each information has a clearly determinable time of origin which indicates, whether the information is relevant now or if it originated earlier and might be obsolete.

3 Elastic Stack

The Elastic Stack is an open source engine capable of reliably importing data from any source and any format to search, analyze and visualize it in near real-time.

The whole stack consists of four main components: Elasticsearch, Beats, Logstash and Kibana [6].

Elasticsearch is a distributed search and analytics engine where all data entries are stored as JSON documents. It does not depend on a specific data source type (structured, semi-structured or unstructured) and it can interface with any RESTful API. Furthermore, Elasticsearch is built on top of Apache Lucene which allows near real-time search capabilities [2].

Logstash is a server-side data processing engine that gathers data from different sources, transforms it and then sends it to an arbitrary output. Logstash is highly customizable through plugins for input, output and data-filters which allow to process a high diversity of data types (e.g. logs, netflow, files) [5].

Beats are lightweight data shippers that are installed on servers to capture operational data of multiple types. They have a small installation footprint, use limited system resources and have no runtime dependencies. Operational data is being sent via Logstash or directly to Elasticsearch. Beats has a complementary role to Logstash - Logstash is a server-side component whereas Beats has a role on the client side [1].

Kibana is the analytics and visualization platform designed to search, view and interact with data stored in Elasticsearch indices. It is possible to present changes in Elasticsearch in near real-time through a variety of visualization types or dynamic dashboards [3].

This can make it easy to under-stand large volumes of data. Combining the four Elastic components, it is possible to create a Big Data analytics solution engine:

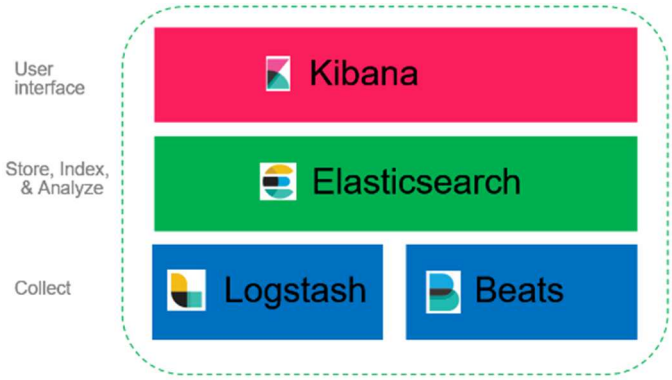

Figure 2: Elastic Stack [11]

4 Visualizing Operational Data

4.1 Scenarios for Enterprise Architecture Design using Operational Data

A proposed scenario is **Correlation Driven Design**, which is based on the high-level idea, that information from operational data can be mapped to quantifiable indicator values which in turn make a certain statement about some component of the Enterprise Architecture. Whether watching these indicators value's increase, decrease or remain static a certain time frame can detect recurring correlations between certain indicators. This could reveal new relationships and interconnections between parts of the Enterprise Architecture which have not been considered before. Additionally, when certain correlation patterns are known, actions or design changes can be applied on the fly, adapting to a critical situation which is indicated by a correlation pattern.

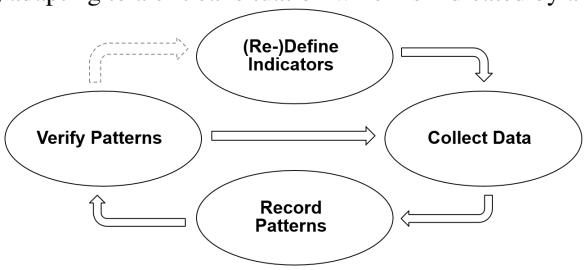

Figure 3: High level Process Sketch

The process begins with the step **Define Indicators,** certain indicator values must be defined which will be subject for later monitoring and investigation. For each indicator value it must be made clear from which part of the architecture it is extracted and how. The indicator must be determined by some output from any readable artifact and the result must be quantifiable within a time frame. Examples for indicators could be the amount of error messages in an application server log, percentage of buffer reads on a database server or the total amount of times where a distinct function of a certain application is called.

Furthermore, it must be clear which part of the architecture the indicator makes a statement about. Like stated in chapter three, the origin of data and the subject can be at different positions of the architecture, even at different layers. An attempt to represent indicators in a design using ArchiMate [8] is shown in Fig 4. In this example it is shown, that the information on how often the Order Process is called might originate from a web server log file.

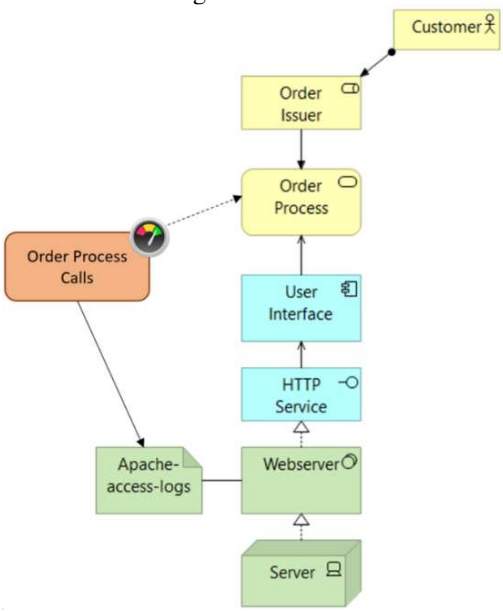

Figure 4: Using Indicators in existing ArchiMate [8] Enterprise Architecture Models

After the indicators have been defined, the step **Collect Data** is about measuring the indicator values by their given input. Since it will be relevant later to record how indicators change over time, it is important to fix a time frame, like "within the last two minutes". This part of the process most certainly requires tool support, especially when real-time interpretation of information from data streams is aspired.

In the process step **Record Patterns,** for each indicator value, it is recorded whether the indicator decreased, increased, or didn't change at all within the recorded time frame. This way, a **Correlation Pattern** is generated which contains the behavior of all indicators in relation to each other. Again, the recording of Correlation Patterns should be automated using software, especially in a real-time scenario.

Before a Correlation Pattern can be used for decision making or in the design process, it must be verified, which is done in the **Verify Patterns** step. By comparing Correlation Patterns to each other, it can be determined, which parts of Patterns reoccur frequently. Depending on the amount of Correlation Patterns recorded and the amount of indicator values, this task should also be supported by software.

After the pattern verification, the process goes back to collect more data, either to find out about new correlations or to verify existing Correlation Patterns. In some cases, it might be necessary to define new indicators or to adjust existing ones. This highly depends on findings made during the analysis process, since readjustment of indicators should lead to clearer and more detailed results.

Another technique described in [4] focused on quantitative input data is **Performance Analysis**, which analyzes the performances of individual components in a system by measuring response times at different levels. Response times on higher layers

can rely on the response times of lower layers which means that the response time of a business service depends on the response time of a related application component which might have a response time depending a server or software response time in the Technology Layer. By modeling a performance view, it is both possible to determine from a top-down perspective, which adjustments need to be done to achieve a determined business service response time, but also to calculate the possible maximum throughput from a bottom-down perspective [4].

It is also stated, that one of the most difficult tasks is to obtain reliable input data. Possible sources can be web server access logs for business response times, as well as application and system software logs which log response times on a more granular level. By aggregating those in a common context, it is both possible to see changes over time but also to use relations between changing factors to find out about new relations.

4.2 Implementing Visualizations in Elastic Stack

With the help of the visualization software Kibana, it becomes possible to visualize data stored in Elasticsearch by a variety of charts, maps or diagrams. To obtain value from the collected data, meaningful dashboards can be built for an effective storytelling.

To make sense out of data and to obtain value, the following data sources have been processed for analysis usage in the shown scenario: Apache2 server access and error logs and performance metrics of the server.

The data sources for these use cases originate from an Elastic Stack server and the test instance of a SWAN server. SWAN is a platform for secure and traceable data exchange developed by SSC-Services GmbH [7].

The whole Project Stack (Figure 5) is based on the following components: The **Elastic Stack server** that includes Elasticsearch (data storage), Kibana (visualization), Logstash (data enrichment), Metricbeat (fetches system metrics) and the **SWAN Stack server** which consists of WildFly (application server), ActiveMQ (Message Broker), Processclient (processing tool), Apache2 (web server), Metricbeat (fetches system metrics), Filebeat (ships logs), Packetbeat (network analyzer).

The SWAN Stack server has 8 GB RAM, 2 CPU(s) and 20 GB HDD.

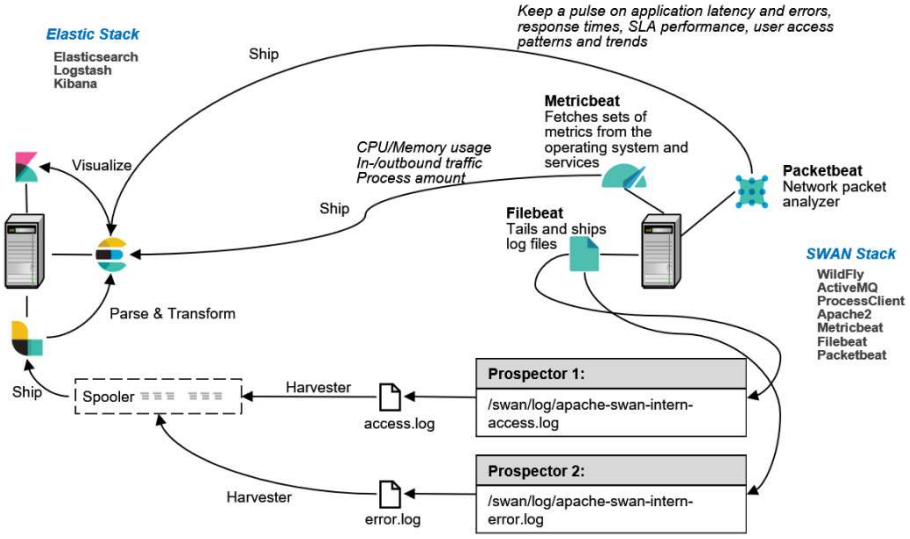

Figure 5: Project Stack Implementation

For our Apache2 access and error log file analysis use cases, the following questions have been raised: What are the top URLs requested? What is the memory usage during the requests?

To get information about the amount of clients requesting a certain URL and how it affects the memory consumption of the whole system, Apache2 access and error logs have been examined.

Figure 6 represents an extract of a time frame showing a significant increase of website calls between 09:35 and 09:45. To get the most valuable information out of the Apache2 log files, a table was created with the following attributes: time, country name, request, response code of the web server, the used browser and operating system of the requestor.

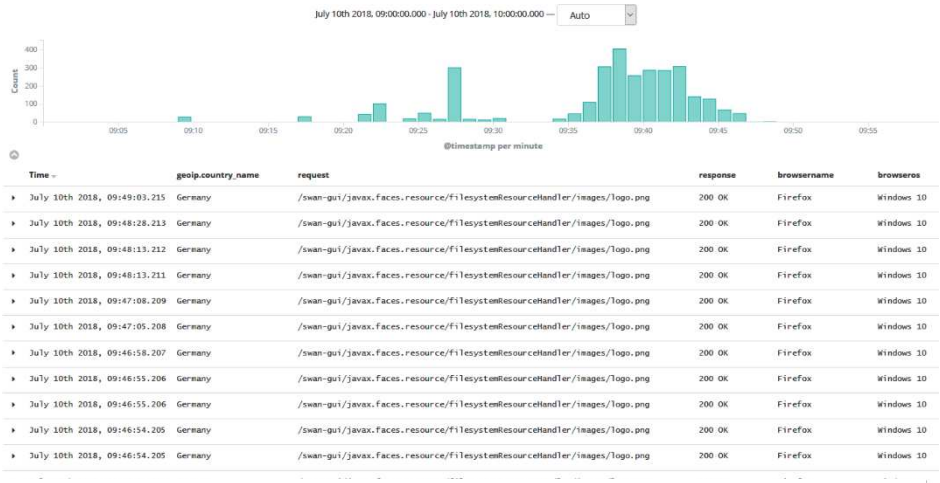

Figure 6: Visualization of Results

At the same time, there is a significant increase of the Apache2 load on the server (Figure 7).

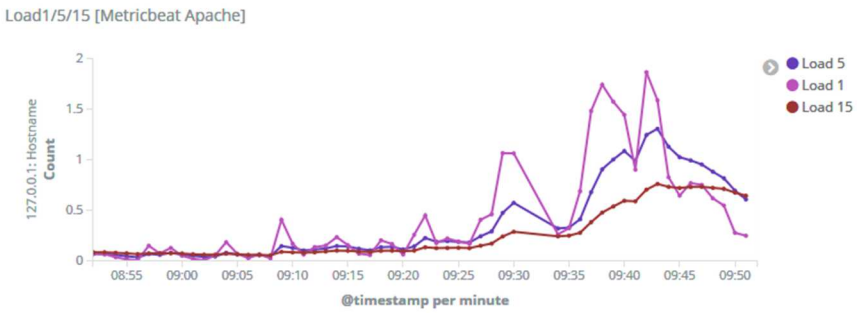

Figure 7: Apache Load

In the same time frame as the request peak appears the performance analysis resulted in Figures 8. The system metrics were collected through the help of the different Beats components. To keep it simple, only a visualization of the system load is shown.

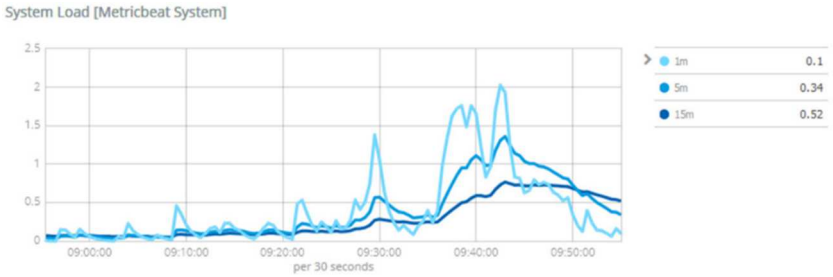

Figure 8: System Load

The whole system load and memory usage increased when the website request peak occurred. All the visualizations can be combined through a dashboard, which makes it easy to search and obtain value from it (Figure 9). At a glance there is all important information. With the help of all visualizations it is much easier to draw conclusion about different coherences of system components which might have been overlooked.

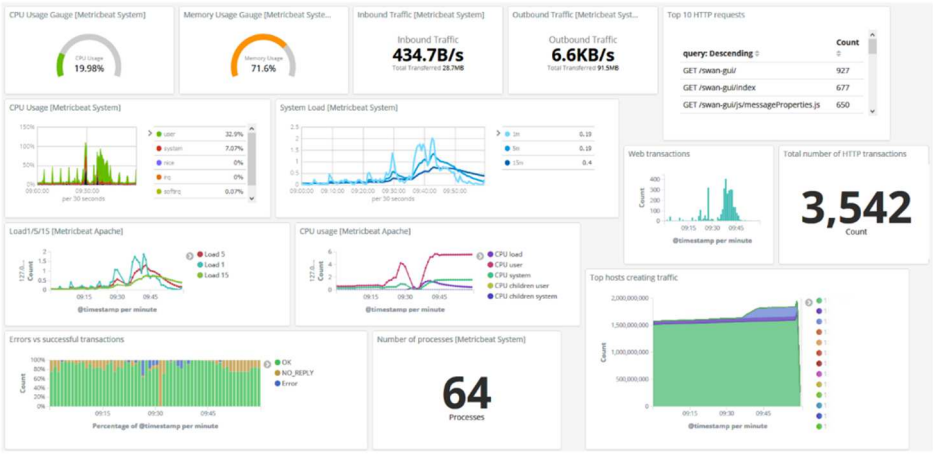

Figure 9: Dashboard

This way, dashboards can be utilized to meaningfully aggregate visualizations which allows a simple way for drawing conclusions about correlations manually. Since Elasticsearch allows processing of diverse data, displayed information can possibly originate from different components as well as different model layers.

5 Related Work

In their research paper “Relating Business Intelligence and Enterprise Architecture”, the authors discuss an approach to use business intelligence solutions to evaluate operational data against architectural metadata [12]. This approach includes a detailed process, which does not involve Big Data tools or real-time analysis and is more focused on the process of the design. Marc Lankhorst’s book “Enterprise Architecture at Work” [13] suggests basic means of modeling, visualizing and working with Enterprise Architectures [13]. Furthermore, it describes processes and analysis scenarios but does not deal with tools or ways to generate input out of operational data. The book “Collaborative Enterprise Architecture” [14] covers everyday processes between stakeholders and users to make design decisions and alterations to an Enterprise Architecture but also

doesn't cover tool-assisted ways to generate input data out of operational information [14]. In most literature about the Elastic Stack, it is mainly suggested to use it to extract, analyze and graphically prepare log files of individual components. Thus, individual components can be monitored in real time [15] [16]. These possibilities provided by the Elastic Stack are what started our project ides: We look at the features of the Elastic Stack and expand it with the following assumption: If operational data has already been prepared for individual components, it is possible to process them to get an overview over all components. Based on this overview, we can recognize connections between components and we can get conclusions about the Enterprise Architecture and visualize them with Kibana.

6 Conclusion

We have shown that there are a several possible scenarios in which the aggregating meaningful visualizations of different types of data from various types of sources in an IT architecture can be used to generate new information which would only be available when all the data is assembled and analyzed in context.

The Elastic Stack provides tools and methods which enables the loading of a variety of diverse input sources and is capable of processing and extracting crucial information from these sources with the goal of visualizing the most important facts in a way that they can be used for decision making in Enterprise Architecture design processes.

The proposed scenario is still quite simple, connecting application log files with infrastructure metrics. The reason for this is, that currently our main goal is to connect Big Data analytics with Enterprise Architecture use cases. As soon as this can be validated in a practical experiment, the scenario can be extended, using other data as well as life data-streams.

However, there would still be a considerable amount of manual work involved: The definition of indicators, the connection of input files and streams to the Elastic Stack system as well as the generation of charts and diagrams would require experts and architects who can take care of the processing. Any findings made by analyzing correlations in Kibana would require manual adjustment at the system landscape.

To automate parts of the decision and adjustment process, it could be a promising scenario to involve machine learning mechanisms which are able to determine necessary adjustments, perform these adjustments and measure the output. This way, it can become possible to automatically gain new knowledge about cause and effect relations which would then be considered in future decisions.

References

1. Beats Platform Reference, https://www.elastic.co/guide/en/beats/libbeat/current/index.html, state: 07.07.2018
2. Elasticsearch Reference, https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html, state: 07.07.2018
3. Kibana User Guide, https://www.elastic.co/guide/en/kibana/current/index.html, state: 07.07.2018
4. Lankhorst, Marc; Enterprise Architecture at Work; 4th Edition, 2017, Springer Verlag, Berlin
5. Logstash Reference, https://www.elastic.co/guide/en/logstash/current/index.html, state: 07.07.2018

6. The Elastic Stack, https://www.elastic.co/de/products, state: 08.07.2018
7. Products SWAN, https://www.ssc-services.de/services/products/swan/?lang=en, state 10.07.2018
8. ArchiMate® 3.0.1 Specification by the Open Group http://pubs.opengroup.org/architecture/archimate3-doc/ , state: 22.01.2018
9. A. Zimmermann, R. Schmidt, K. Sandkuhl, D. Jugel, J. Bogner, M. Möhring, "Decision Management for Micro-Granular Digital Architecture," In S. Hallé, R.M. Dijman, J. Lapalme (Eds.) EDOC / SoEA4EE 2017, IEEE, pp. 29-38, 2017
10. Open Group, "TOGAF Version 9.2," The Open Group, 2018.
11. Starte mit Kibana, https://www.elastic.co/de/webinars/getting-started-kibana?view=1, state: 05.07.2018
12. Veneberg, R.K.M; Iacob, M.E.; van Sinderen, M.J.; Bodenstaff, L.; Relating Business Intelligence and Enterprise Architecture, 2017, Enschede, World Scientific Publishing Company
13. Lankhorst, Marc; Enterprise Architecture at Work; 4th Edition, 2017, Springer Verlag, Berlin
14. Bente, Stefan; Bombosch, Uwe; Langade Shailendra; Collaborative Enterprise Architecture; 2012; Morgan Kaufmann; Waltham MA
15. AL-Mahbashi, Ibrahim Yahya Mohammed; Potdar, M. B.; Chauhan Prashant (2017): Network Security Enhancement through Effective Log Analysis Using ELK. In: Proceedings of the IEEE 2017 International Conference on Computing Methodologies and Communication (978-1-5090-4890-8/17/), S. 566–570. Online available under https://ieeexplore.ieee.org/document/8282530/, state 10.07.2018
16. Bai, Jun (2013): Feasibility Analysis of Big Log Data Real Time Search Based on Hbase and ElasticSearch. In: Ninth International Conference on Natural Computation (ICNC), S. 1166–1170. Online available under https://ieeexplore.ieee.org/document/6818154/, state 10.07.2018