Scalable Sandbox Environments for a Modern Organization

Artyom Topchyan <a.topchyan@reply.de>
Yerevan State University,
Alek Manukyan 1, Yerevan, 0025, Armenia

Abstract. With the growing volume and demand for data, a major concern for an Organization is the creation of collaborative Data-Driven projects. With the amount of data, number of departments and the development of potential use cases, the complexity of creating Multitenant and Collaborative environments for multidisciplinary teams to work and create productive solutions, is becoming more and more important problem. In this work, we describe an approach to building such an environment with scalability, flexibility and productivity in mind. This solution is an integral part of a joint Operational Data Platform for data exploration and processing at large data driven Organizations.

1. Introduction

With the growing volume and demand for data a major concern for an Organization is to find a well-defined way of enabling large teams to work collaboratively on a large amount of very varied data. This trend is driven by business and technical demand, which especially stems from the need for more Data-Driven projects [1][3][4]. Data-Driven projects aim at increasing the quality, speed and/or quantity of information gained from Data collected by the Organization. But it is very challenging to move ahead with such projects without the need to define models to handle data exploration and processing. As we described in our previous work [1][2], this leads to most of the project time being spent not on data analysis, but finding out information about the existing data, getting access to a usable form of the data and requesting a suitable ICT environment to process this data. This can be quite costly time and resource wise, and each stage of a Data-Driven project is impacted by this.

There is currently no single accepted approach for tackling such problems, so we described and implemented a joint Operational Data Platform(ODP) for data exploration and processing. In [1] we described the building blocks of the platform and went into detail, how the data is collected and stored. In [2]. we described the Information Marketplace, which is an intelligent search engine system, specifically designed to tackle the problem of information retrieval and sharing in a large multifaceted organization, that already has many systems in place for each Department. This platform has been successfully implemented and is actively used by large
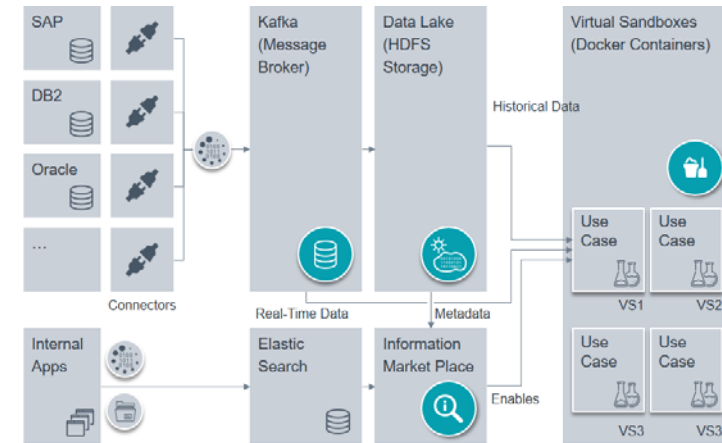

Fig. 1. Operational Data Platform

organizations to implement Data-Driven projects. A high level overview of the entire solution as presented in our previous work is outlined in Fig.1. In this work we will outline the third component of the Platform, which centers on providing a scalable environment to allow for collaborative and multi-developer projects. This approach aims to simplify testing and deployment in order to bring these highly complex projects into production. We will first outline why we believe such a solution is required. Next we will define a set of requirements, such a solution must fulfill. We will then outline the architecture, that aims to address these requirements, as well as how it differs from commonly accepted approaches and the various technical challenges involved in building such a system.

2. Data-Driven project development environment

As highlighted in [1] most Data-Driven projects in the industry often follow a variation of the Cross Industry Standard Process for Data Mining project lifecycle, as described in [5]. Which states that most of these project go through the following steps:

- Business Understanding;

- Data Understanding;
- Data Preparation;
- Modeling;
- Evaluation;
- Deployment.

In [1] we outlined how we address the necessity for enabling and optimizing these individual steps and in [2] we give more detail how to take step one and two and greatly simplify step 3 and 4. But we did not fully consider yet, is in what what context this project is developed. Projects are developed in environments, which first of all usually reflects internal requirements of the project. In our case project vary greatly in their complexity and the amount of data they process. So a environment is required that can generalize their parameters to the main functional requirements of the problem. A very important point to note in this context is that most Organizations require for these project to be developed in a a fully on-premise secure environment, which greatly limits what can be done towards building a scalable solution. This means we need to design the environment to fit the largest possible problem, that would need to be solved, which entails processing terabyters of data from many disparate systems. To this end one practical solution used in the industry is building clusters of many machines, that can handle these types of workloads [7] [8] In a modern Organization this usually means using the Hadoop ecosystem [6] to create a generalized processing environment with support for most data types and approach to analysis. This usually goes hand in hand with a Data Lake implementation [9] or alternatively a more flexible and structured approach as described in [1] Hadoop clusters are immensely powerful and flexible, but suffer from a number of oversights. Hadoop environment usually do not provide a very flexible way for dozens of users to access the systems in parallel. The usual approach for this is to provide an Edge Server [8], which is a secured server, that allows for selective access to certain resources of the cluster. There are usually a set number of such server per cluster and are not necessarily very powerful servers. The idea behind such servers is, that they should only be used to deploy job to the cluster and not carry out any calculation themselves. This is more geared towards Data-Engineers deploying applications and not an analysis environment, where Data Scientists have to explore the data and retrieve results in an ad-hoc manner. For this analysis many routine libraries and tools are often required and this leads to the Edge Servers running out of resources and contain a large number of potentially clashing software as these are shared environments. As more and more users start using these servers this becomes an impractical approach. To address this there are approaches to use vitalization to provide on demand environments for each individual team [11] [10] Due to vitalization overhead multiple teams need to share these machines, which are often over-provisioned and have much more resource than it is required as reconfiguring such machines is usually a lengthy administrative task. Requiring multiple teams to share a virtual environment is also the simplest way to enable collaboration as

individual developers can access the code, results or services of other team members. But this leads to the resource of the machines not being fully utilized and decreased performance when the project requires a large number of services or processing workloads, that do not run on the cluster. For example, prototyping a system similar to the Information Marketplace to analyse and index the documents and data of the Organization, would require, a Search engine, a Database and many other services. Another downside to such approaches is the complexity it brings to testing and deploying such projects, once development has finished. It is unclear what libraries and resources are required to actually run the application outside the development environment. Dependencies, configurations and resource specifications have to be supplied to the operations team, which has to somehow package this into a solution that conforms to the organizations standards and can be run in production. Such Edge Servers are not self-describing and fully rely on the development team to deliver well documented, testable, deployable code. This is often not simple to do due to the multidisciplinary and complex nature of Data-Driven projects. What is required is a self-describing environment, that is simple to ship into production. It is possible to directly deploy the vitalized edge-nodes with all the required software inside them, but this is quite expensive and slow, when working in an on premise environment as such approached are too rigid in their requirements. To summarize let's pose a list of requirements we accumulated based on our experience working with teams of Data Scientists at large organizations:

1. Single-User Secure isolated environments. It is beneficial for users to have fully isolated environments as this simplifies development and later on deployment. Because of this environment need to be fully isolated and secured in accordance with the Organizations requirements.

2. Support a large variety of demanding workloads. Such environment should support all types of workloads, that can be submitted to the Hadoop cluster as well as ran locally inside the environment itself, which is essential for ad-hoc analysis. Such environments should also support being used as servers for tools required by other projects. For example, a standalone Database deployment.

3. Managing Resources in a highly multi-tenant environment. We need flexibility to manage many of such instances efficiently, potentially on very limited hardware. Environments should consist of services based on required resource allocation

4. Collaboration tools.The environment should have in-built tools that enable collaboration. This is very important to enable well documented, testable projects and code.

5. Scalability and Fault-Tolerance. It should be straightforward to scale individual environments up and down on demand, in perspective

automatized. As it is important to ensure uninterrupted work and no loss of work data, such environment should be Fault-Tolerant.

6. Going to Test, Production quickly after development. The environments should be self-descriptive and flexible enough to speed up this process and not introduce more technical hurdles.

7. Flexible access to all stored Data. Because this is an environment for a Data-Driven project, the most important thing is access to all required data, but this access has to be customizable as to provide fun grained access for teams and individual developers.

This is not an exhaustive list, but it outlines the basic requirements such a system must fulfill from the view point of teams working on Data-Driven projects. It is list of complex functional requirements which Project, System Developers and Designers need to map to a technical problem definition and implementation. To this end we propose the analytical sandbox environments, which are generated, isolated environments provided to Data Scientists, Analysts and Engineers so that they can build up their project on the data. It is a fully isolated environment, where the user can install or download any extra tools they require and is accessible via an analytical and console view.

3. On-demand Sandbox environments

There are significant architectural and algorithmic considerations when mapping the requirement set outlined in the previous chapter to a technical implementation. We will outline the technical design of these sandboxes and the necessary components we use to provide the technical functionality.

3.1 Immutable environments

First of all we require access to all the existing cluster resources preferable via the native interfaces, but provide a remote accessible and simple to use environment. As one of the main gaols is to simplify development, this approach should also support some way to distribute source code and dependencies along with the environment. To summarize we require this environment to be stateless and immutable as possible, but still contain all the necessary tools and code when moved. To achieve the flexibility required we just replace it with another instance to make changes or ensure proper behavior. This would allow sandboxes to not only serve as an analytical environment, but as a template for all Data-Driven projects, that ever come to production. To this there has been a large shift in the Industry to use Linux Containerization technology [12] to solve these types of problems.

3.1.1 Linux Containers

LXC (Linux Containers) is an operating-system-level virtualization method for running multiple isolated Linux systems (containers) on a control host using a single

Linux kernel [12] The Linux kernel provides the cgroups functionality, that allows limitation and prioritization of resources (CPU, memory, block I/O, network, etc.) without the need for starting any virtual machines, and namespace isolation functionality that allows complete isolation of an applications' view of the operating environment, including process trees, networking, user IDs and mounted file systems [12] Such environments can be created based on a defined specification, which allows the environment to inherently be self-describing [12] This essentially would allow us to launch anything we want in completely independent environments. For example, on-demand edge nodes. This approach is very useful for creating immutable architectures, where every component can be replaced at any point in time. It is being used more and more in the Industry and is becoming the standard for running distributed services and applications [15] The use of this type of process management and assuming that each sandbox is fully isolated and immutable allows us to easily reason about Fault-Tolerance and Scalability.

3.2 Fault Tolerance and Scalability

In the context of Immutable containers, service fault tolerance and scaling becomes a process scheduling problem. Technically we have a set amount of resources in total. Each sandbox takes a certain amount of resources, such as CPU, RAM, Disk space and Network. Considering the fact, we need a large amount of resources, but limited amount of Hardware at our disposal, we need to effectively plan where each sandbox can run and how much resource it can actually use. This is a well-known problem in the Industry as is usually tackled by global Resource Manager or planners, such as Apache Mesos [13], Google Kubernetes [16] or Borg [14] and there are well defined patterns of tackling such problems [15] In this work we use Apache Mesos due to its popularity in the Industry and its proven ability to scale. Such a resource manager is essentially a higher level version of an Operating Systems Kernel running on many machines. Applications decide what resources they require based on their processing time and requirement, and the resource manager tries to accommodate this. If the service crashes or the computing node fails, it can be transparently restarted somewhere else assuming the application can transparently handle something like this. This would allow us to efficiently run many sandboxes on very limited hardware by automatically scaling how much resource each one requires as well as making each sandbox Fault-Tolerant by launching another one during outages.

3.3 Collaboration

A central aspect of a solution like this is simplifying collaboration. As we propose isolated environments, this becomes more challenging. The accepted approach using central source code-based collaboration, such as Git and SVN [21], which we also adopt. But it is often challenging to share large files, such as models, test data and dependencies between team members using these systems. To this end we implement a shared network file-system layer between team members of the same project. Each project team gets a filesystem, with individual workspaces for each members. Team

members can then decide how to structure this environment in order for it to be more productive for them. One crucial requirement for such system is Fault-Tolerance. Which is even more crucial due to our implantation of Immutable sandboxes. Any large files generated by the project team should be durable to sandbox and computing node outages. Loosing modelling results can lead to days of lost work time. To address this there are a number of high performance and durable implantation of Network Filesystems being used in the Industry [22] [23] In this work we adopted GlusterFS as our network storage due to its proven performance, simplicity and integration capabilities with systems like Mesos [23]

3.4 Isolation

As we outlined above isolating the sandboxes simplifies management and development. By isolation we are specifically referring to isolation of resources, such as:

- CPU
- RAM
- Filesystem
- Libraries and System tools

Most of these are out-of-the box supported by Linux container technology and are supported and managed by Apache Mesos. On the other hand, as these sandbox environments are not exclusive to Analytical Cases, which produce reports as their output, but Service based applications as well. An example of this would be a Web Service that provides recommendations based on customer input data. Such applications usually run as a web service and most commonly rely on some database system, which might not be provided as part of the Hadoop installation. Such services can be easily launched in a separate sandbox environment. We provide teams of developers with easy access to these services or databases, provided they are part of the same use case. As there might be any number of such services running inside the cluster, this might lead to problems as defined in [20] So in order to simplify the development process we would need full Network isolation as well. To solve this, it is now an accepted approach in the Industry to use software based overlay network solutions, which are computer networks that are built on top of another network [18] [19] This gives us the ability to selectively isolated certain Sandboxes from each other, allowing the users of these sandboxes to launch any service they want in their sandboxes and transparently make it accessible to other team members, without leading to network clashes with other environments. In this solution we use Calico networking, which is well integrated with Mesos environments.

3.5 Security

Due to the level of isolation provided Security becomes an Authorization problem and can be implemented as per requirements from the Organization. A commonly accepted approach in the Industry is to integrate with a central Organization wide

authority, which has all the information about user rights and permissions. In most cases we integrate directly with the Organizations central LDAP, such as Kerberos or Microsoft Active Directory [17] As most Hadoop installation also support this integration we can transparently enable security throughout the solution without introducing a new authorization and authentication concept in the Organization.

4. Architecture

Having discusses the individual building blocks of the solution, we will outline how this translates a technical architecture and implementation as a number of automation services are required to build the necessary platform based on the concepts outlined in the previous section. This architecture is implemented similarly to our previous work and must fulfill the requirements of low-latency and flexible services. To this end we adopted a Microservice architecture, which a lightweight and flexible approach to implementing Service Oriented Architectures [24] The Sandbox orchestration service is defined with the following components.

4.1. Sandbox Templates

The first required component is a set of templates for sandbox environments. These vary based on the Organizational standards, but these typically contain all the necessary tools to interface with the Hadoop Cluster, run analytics and build applications. For example:

- Secure Shell access
- Ipython Console, R console, Scala Console
- Hive, Hue, Hadoop
- Pyspark, Spark, RSpark

These are common tools and having a repository of such as propose built templates allows to formalize the tool choice and testing and deployment processes during development. New templates can be added based on demand by extending the already existing ones.

4.2 Marathon

Marathon is a production-grade container orchestration platform for Apache Mesos [25] This allows for API based orchestration of containers based on the specific templates. We use this as our central orchestration platform for all sandboxes.

4.3 Kontrollores

Kontrollores is our central control service. It handles request for creating new teams and sandboxes for them. It provides the following functionality:

- Handle Sandbox creation requests;
- Authorize request;

- Create code repositories;
- Create collaboration file system;
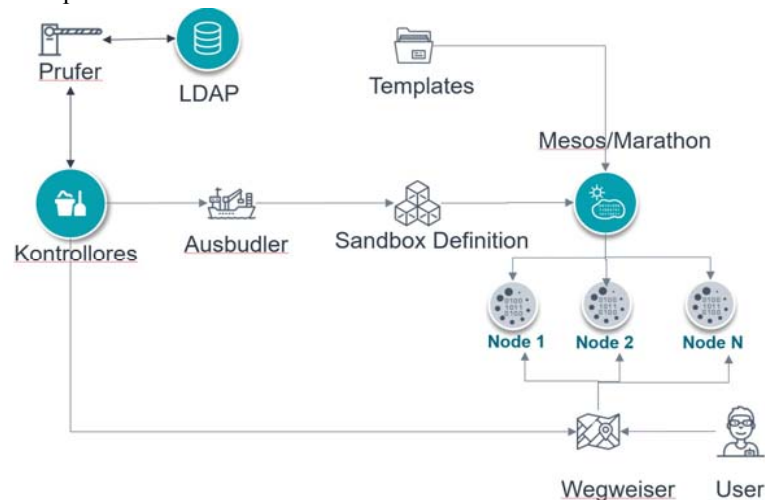- Initiate Sandbox creation;
- Report status.

Fig, 2. Sandbox microservice architecture

4.4 Ausbudller

Ausbudller Is an interface service to Marathon and handles the actual creation and monitoring of a single sandbox. It validates these request based on the user permissions and quotas stored in the central LDAP.

4.5 Prufer

Prufer handles validation of resource and tools requests based on the users in the team.

4.4 Wegweiser

Wegweiser is the central routing component. As the sandboxes are in isolated environments we require a secure gateway to dynamically route individual users through the internal network to their sandbox. This is achieved by discovering sandbox details based on their definition retrieved from Kontrollores.

The combined architecture defined with these components is outlined in Fig, 2, which also outlines the interconnection of these components and the general flow of execution.

5. Conclusion

We proposed an approach for a scalable, multi-user environment for handling research and development of Data-Driven projects in the context of a large organization operating a Hadoop cluster. We described the necessity for such a solution to handle the complexity of managing multi-user research projects and outlined the technical challenges faced when implementing similar systems. Based on this we outlined the set of requirements desired by many Organizations in the industry and we proposed a scalable, fault-tolerant and flexible Architecture as well as technical implementation, that satisfies all these requirements. We believe the approach, proposed solution and its architecture provide a solid basis for implementing similar systems at large Organizations, that are starting to explore Data-Driven projects.

References

[1]. Topchyan A.R. Enabling Data Driven Projects for a Modern Enterprise. Trudy ISP RAN/Proc. ISP RAS, vol. 28, issue 3, 2016, pp. 209-230. DOI: 10.15514/ISPRAS-2016-28(3)-13

[2]. Topchyan A.R. Information Retrieval and Analysis for a Modern Organization. *Trudy ISP RAN/Proc. ISP RAS*, vol. 28, issue 4, 2016, pp. 7-28. DOI: 10.15514/ISPRAS-2016-28(4)-1

[3]. Rahman, Nayem, and Fahad Aldhaban. "Assessing the effectiveness of big data initiatives."2015 Portland International Conference on Management of Engineering and Technology (PICMET). IEEE, 2015.

[4]. Davenport, Thomas H., and Jill Dych´e. "Big data in big companies."International Institute for Analytics (2013).

[5]. Shearer C. The CRISP-DM model: the new blueprint for data mining. J Data Warehousing; 5:13—22, 2000

[6]. Tom White. Hadoop: The definitive guide. O'Reilly Media 2012

[7]. Rowstron, Antony, et al. "Nobody ever got fired for using Hadoop on a cluster. "Proceedings of the 1st International Workshop on Hot Topics in Cloud Data Processing. ACM, 2012.

[8]. Lin, Chia-Feng, et al. "The study and methods for cloud based CDN."CyberEnabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on. IEEE, 2011.

[9]. Alex Gorelik. The Enterprise Big Data Lake: Delivering on the Promise of Hadoop and Data Science in the Enterprise. O'Reilly Media 2016

[10]. Shen, Zhiming, et al. "Cloudscale: elastic resource scaling for multi-tenant cloud systems. "Proceedings of the 2nd ACM Symposium on Cloud Computing. ACM, 2011.

[11]. Li, Peng, Lee Toderick, and Joshua Noles. "Provisioning virtualized datacenters through virtual computing lab."2010 IEEE Frontiers in Education Conference (FIE). IEEE, 2010.

[12]. Rami Rosen. "Resource management: Linux kernel namespaces and cgroups"(PDF). cs.ucsb.edu. Retrieved February 11, 2015.

[13]. Apache Mesos. http://mesos.apache.org, 2015.

[14]. Verma, Abhishek, et al. "Large-scale cluster management at Google with Borg."Proceedings of the Tenth European Conference on Computer Systems. ACM, 2015.

[15]. Burns, Brendan, and David Oppenheimer. "Design Patterns for Containerbased Distributed Systems."8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16), 2016.

[16]. Burns, Brendan, et al. Borg, Omega, and Kubernetes."Communications of the ACM 59.5: 50-57, 2016.

[17]. Koutsonikola, Vassiliki, and Athena Vakali. "LDAP: framework, practices, and trends."IEEE Internet Computing 8.5: 66-72, 2004

[18]. Project Calico. https://www.projectcalico.org 2015.

[19]. Jain, Raj, and Subharthi Paul. "Network virtualization and software defined networking for cloud computing: a survey." IEEE Communications Magazine 51.11: 24-31, 2013

[20]. JAMES, BD. "Isolating network traffic in multi-tenant virtualization environments."U.S. Patent No. 7,885,276. 8 Feb, 2011.

[21]. Otte, Stefan. "Version Control Systems."Computer Systems and Telematics, 2009.

[22]. FraunhoferFS "FraunhoferFS High-Performance Parallel File System "Computer Systems and Telematics, 2009.

[23]. Klaver, Jeroen; van der Jagt, Roel. Distributed file system on the SURFnet network Report, University of Amsterdam System and Network Engineering, 2010

[24]. Newman, Sam. Building Microservices. "O'Reilly Media, Inc.", 2015.

[25]. Marathon. https://mesosphere.github.io/marathon, 2015.

Масштабируемые учебно-экспериментальные среды для современных предприятий

А.Р. Топчян <a.topchyan@reply.de>
Ереванский государственный университет,
0025, Армения, г. Ереван, ул. А. Манукяна, дом 1

Аннотация. С ростом объема данных и потребностей в них одной из основных проблем организаций становится создание кооперативных, управляемых данными проектов. При возрастании объема данных, числа подразделений и появлении новых потенциальных сценариев использования все более важным является создание корпоративных сред, поддерживающих совместную работу мультидисциплинарных групп для разработки эффективных решений. В этой работе мы описываем подход к построению такой среды, обладающей масштабируемостью, гибкостью и производительности. Это решение является неотъемлемой частью совместной оперативной платформы данных по изучению и обработке больших данных в крупных организацией, ориентированных на обработку данных.

Ключевые слова: проекты, ориентированные на данные; учебно-экспериментальные среды; автомасштабирование; кооперация; отказоустройчивость; изоляция; контейнеры; Mesos; микросервисы.

Список литературы

[1]. Topchyan A.R. Enabling Data Driven Projects for a Modern Enterprise. Trudy ISP RAN/Proc. ISP RAS, vol. 28, issue 3, 2016, pp. 209-230. DOI: 10.15514/ISPRAS-2016-28(3)-13

[2]. Topchyan A.R. Information Retrieval and Analysis for a Modern Organization. *Trudy ISP RAN/Proc. ISP RAS*, vol. 28, issue 4, 2016, pp. 7-28. DOI: 10.15514/ISPRAS-2016-28(4)-1

[3]. Rahman, Nayem, and Fahad Aldhaban. "Assessing the effectiveness of big data initiatives."2015 Portland International Conference on Management of Engineering and Technology (PICMET). IEEE, 2015.

[4]. Davenport, Thomas H., and Jill Dych´e. "Big data in big companies."International Institute for Analytics (2013).

[5]. Shearer C. The CRISP-DM model: the new blueprint for data mining. J Data Warehousing; 5:13—22, 2000

[6]. Tom White. Hadoop: The definitive guide. O'Reilly Media 2012

[7]. Rowstron, Antony, et al. "Nobody ever got fired for using Hadoop on a cluster. "Proceedings of the 1st International Workshop on Hot Topics in Cloud Data Processing. ACM, 2012.

[8]. Lin, Chia-Feng, et al. "The study and methods for cloud based CDN."CyberEnabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on. IEEE, 2011.

[9]. Alex Gorelik. The Enterprise Big Data Lake: Delivering on the Promise of Hadoop and Data Science in the Enterprise. O'Reilly Media 2016

[10]. Shen, Zhiming, et al. "Cloudscale: elastic resource scaling for multi-tenant cloud systems. "Proceedings of the 2nd ACM Symposium on Cloud Computing. ACM, 2011.

[11]. Li, Peng, Lee Toderick, and Joshua Noles. "Provisioning virtualized datacenters through virtual computing lab."2010 IEEE Frontiers in Education Conference (FIE). IEEE, 2010.

[12]. Rami Rosen. "Resource management: Linux kernel namespaces and cgroups"(PDF). cs.ucsb.edu. Retrieved February 11, 2015.

[13]. Apache Mesos. http://mesos.apache.org, 2015.

[14]. Verma, Abhishek, et al. "Large-scale cluster management at Google with Borg."Proceedings of the Tenth European Conference on Computer Systems. ACM, 2015.

[15]. Burns, Brendan, and David Oppenheimer. "Design Patterns for Containerbased Distributed Systems."8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16), 2016.

[16]. Burns, Brendan, et al. Borg, Omega, and Kubernetes."Communications of the ACM 59.5: 50-57, 2016.

[17]. Koutsonikola, Vassiliki, and Athena Vakali. "LDAP: framework, practices, and trends."IEEE Internet Computing 8.5: 66-72, 2004

[18]. Project Calico. https://www.projectcalico.org 2015.

[19]. Jain, Raj, and Subharthi Paul. "Network virtualization and software defined networking for cloud computing: a survey." IEEE Communications Magazine 51.11: 24-31, 2013

[20]. JAMES, BD. "Isolating network traffic in multi-tenant virtualization environments."U.S. Patent No. 7,885,276. 8 Feb, 2011.

[21]. Otte, Stefan. "Version Control Systems."Computer Systems and Telematics, 2009.

[22]. FraunhoferFS "FraunhoferFS High-Performance Parallel File System "Computer Systems and Telematics, 2009.

[23]. Klaver, Jeroen; van der Jagt, Roel. Distributed file system on the SURFnet network Report, University of Amsterdam System and Network Engineering, 2010

[24]. Newman, Sam. Building Microservices. "O'Reilly Media, Inc.", 2015.

[25]. Marathon. https://mesosphere.github.io/marathon, 2015.