

Exploring DevOps for Data Analytical System with Essential Demands Elicitation

Jiabin ZHENG*, Yan LIU†

School of Software Engineering, Tongji University

Shanghai, China

Email: *1434321@tongji.edu.cn, †yanliu.sse@tongji.edu.cn

Jin Lin‡

Shanghai Maitrox Electronics Co., Ltd

Shanghai, China

Email: ‡jeffrey@maitrox.com

Abstract—DevOps is an emerging concept and methodology for bridging the gap in the process of software development. At present, applying DevOps to data analytical system (DAS) is increasingly embraced. But the characteristics of this system, such as data protection, always leads to a series of constrains. It's a bit difficult to conduct DevOps on data analytical system. Moreover, there are no DevOps solutions for reference. Therefore, exploring DevOps for data analytical system is valuable.

In this paper, we illustrate DevOps demands of data analytical system from different perspectives, and constantly emphasize the importance of automation toolchain. Based on them, a process model for DAS DevOps (D^2Ops) is proposed to clarify participants activities. In order to improve the efficiency, we attempt to integrate the automation toolchain. With the consideration of stability, six generic process components are designed to support this model. They can be the selection criteria for specific automation tools. We also present a reference facility based on these generic process components, and illustrate its implementation combing with a practical case.

Keywords—DevOps; Data Analytical System; Constraint; Automation Tool; Process Component;

I. INTRODUCTION

Along with the vigorous development of Internet, large amounts of data are produced daily in various fields. For extracting valuable informations from raw data, the data analytical system has become a rising trend nowadays. However, it probably has a more complex process and more different participants compare with general system. To reduce the difficulty of system development, DevOps, an emerging concept and methodology, is brought in to optimize process.

DevOps originally derives from the demands of solving the misunderstanding and collision between development and operations. Now it involves with not only development and operations, but also integration, delivery, quality assurance and other relevant aspects[1]. DevOps describes practices that streamline the software delivery process, emphasizing the learning by streaming feedback from production to development and improving the cycle time[2]. At present, numerous open-source automation tools have been supported this rising concept. A few practical successful cases also demonstrate its significance. As a result, effective DevOps can be regarded as a promising way for data analytical system.

But there are inevitable challenges. They will be a great problem for carrying out DevOps.

- *No solutions for reference* - In essence, DevOps is only a conceptual principle. It doesn't provide any practice instruction. Consequently, the lack of standardization results in no common solutions. Almost all of the existing DevOps solutions are conducted empirically according to technology actuality. It's hard to take them for reference.
- *Constraints of system characteristic* - Data analytical system is quite different from other systems. It should put great efforts on data protection. Moreover, the result of data analysis is influenced by various factors. These system characteristics lead to constraints that may break the normal DevOps process.
- *Complexity of tool integration* - Massive open-source automation tools will be applied to DevOps practice. But there are too many various tools in the open source community. How to pick appropriate tools and how to integrate them into a full toolchain are bothering people.

To overcome obstacles brought by these challenges, our research focuses on exploring DevOps for data analytical system. The contribution includes:

- Analyze DevOps demands for data analytical system
- Propose a reference process model to clarify DevOps participants and their collaboration
- Design six generic components to support the envisaged DevOps process model

The remainder of this paper is structured as follows. In Section II we conclude the related achievements about DevOps in both of academia and industry. The fundamentals of our research are shown in Section III. We analyze the D^2Ops demands of different perspectives. With the consideration of DevOps participants, Section IV and Section V propose a reference process model. Meanwhile, six generic process components are designed to support this model. Based on them, we demonstrate a reference facility and analyze its essential workflows. In Section VI, we present the facility implementation combined with a practical case. Finally, in Section VII, we illustrate the shortages of our research and make a plan for future work.

II. RELATED WORK

A. Academic Research

Walls[3] encouraged to build DevOps culture, and illustrated the key aspects of promoting success. By applying a qualitative analysis approach, Bang et al.[4] investigated the supporting way of factors, like knowledge, skills and abilities, for DevOps. Lwakatare[5] identified four main dimensions of DevOps: collaboration, automation, measurement and monitoring. Moreover, he developed a conceptual framework to facilitate the understanding of DevOps phenomenon.

It's apparent that all the mentioned above belong to conceptual researches that emphasize on culture propagation and conduction principles. But except those conceptual researches, achievements related to practice shouldn't be neglected. For example, Stillwell et al.[1] summarized their DevOps approach in HARNESS, a EU research project. Based on DevOps, Ferry et al.[6] proposed a continuous deployment solution for multi-cloud systems.

While, by investigation, vast majority of academic achievements are conceptual until now. As the lack of standard practice solutions and metrics for reference, DevOps practice is usually in a mess and hardly estimates whether it is reasonable.

B. Industrial Exploration

Now organizations in industry also constantly propose DevOps solutions for benefiting from them[7]. For instance, PayPal proposed a solution for continuous integration in the Hybrid Cloud. Cisco designed a solution in a SDN world. IBM implemented a generic solution of continuous delivery. While most of them are based on organization's technology actuality regardless of universality. No two enterprise-adopted DevOps approaches would be similar as each enterprise has unique characteristics and requirements[8]. Thus it's a bit difficult to apply these industrial DevOps solutions for us. Now this problem has been gradually realized. Some corporations like Microsoft, IBM and Amazon, attempt to establish common integrative DevOps solutions[9][10][11]. But these solutions are mostly based on their own cloud platforms that are inappropriate for confidential data.

C. Innovative Achievement

Apart from previous researches, now researchers have innovatively tried to combine DevOps with other fields. It's no longer confined to software development process. Babar et al.[8] utilized the technology of Business Process Architecture modeling to select customized DevOps processes. Bruneo et al.[12] used DevOps concept to develop an adaptive cloud management platform. Kim et al.[13] even discussed the DevOps requirements specific to the modern network service creations, and proposed an extended DevOps concept.

III. DEMAND ANALYSIS FOR D²OPS

A. Functional Demand

As similar with general DevOps solutions, the key points of functional demand for D²Ops include: (a) cost-saving; (b) automation; (c) the integrated toolchain; (d) data governance.

According to the statistics[14][15], DevOps has a remarkable effect on slashing cost. To a large degree, the effect is established on automation mechanism. A major advantage of automation is that tasks can be executed and information can be collected relatively more accurately and quickly as compared to manual approaches[16].

The implementation of automation requires a strong supporting from tools[17]. But processes are more important than tools for DevOps[2]. If applied in complete isolation, tools couldn't play their roles effectively. Then it's rather beneficial to combine processes with tools. Generally speaking, toolchain is a collection that organizes tools in accordance with a specific process. By contrast with the way of gathering tools simply, the integrated toolchain is more able to contribute to immense automation effects.

In addition, data quality needs major concerns. It's usually awful because data often come from various sources. The bad data quality may lead to a failure of constructing data models, and even disable the development process. Therefore, data governance technology is needed to ensure good data quality.

B. Premised Constraints

There are several special characteristics in data analytical system. On one hand, people are obliged to put enormous energy on data protection. On the other hand, the result of data analysis is influenced by many factors. These characteristics are the premises of data analytical system. They will result in some constraints.

- *Network Accessing Control* - Network is one of the most insecure tools now. Hence, network accessing control must be reinforced. For this purpose, a simple solution is the setup of LAN (Local Area Network) environment. This solution is often applied as a basic protection for data center. It aims to cut all channels to the Internet.
- *Data Accessing Restriction* - In order to limit the behavior of arbitrary data accessing, production environment and development environment should be separated. This isolation strategy is able to prevent the real data from being accessed outside. Furthermore, developers can still utilize the Internet resources.
- *Data Information Protection* - In the process of development, it's usually unavoidable to take advantage of third-party platforms. However, data-related informations, like source code, are able to reflect real data contents. Once informations are uploaded to those platforms, it may lead to confidential data leakage indirectly. Thus data informations should be strictly controlled.
- *Traceability of Analysis Result Quality* - Data analytical system depends heavily on visualization to represent data analysis result. An unexpected result is probably caused by various factors, like wrong visualization implementation, wrong data model implementation or wrong data model. It's hard to trace the root cause of quality issues.

As TABLE I shows, these constraints will bring about some latent issues. Although common solutions are able to cope with

TABLE I. Constraints Analysis for D^2Ops

Constraint	Latent Issues	Common Solutions	DevOps Enhancement
Network Accessing Control	<ol style="list-style-type: none"> 1. Cannot depend on web-based services in production system; 2. Cannot utilize public repositories on the Internet for server management; 	<ol style="list-style-type: none"> 1. Abandon code packages that need to connect with Internet; 2. Manually install tools and configure without Internet; 	<ol style="list-style-type: none"> 1. Set up local code package repositories like Nexus; 2. Build local tool repositories like local yum and local apt-get;
Data Accessing Restriction	<ol style="list-style-type: none"> 1. Cannot construct and verify data models in development environment; 2. Disable continuous deployment process; 	<ol style="list-style-type: none"> 1. Generate test data based on real data; 2. Manually deploy by transmission medium; 	Utilize virtualization technology for deployment
Data Information Protection	Cannot upload data-related informations to third-party platforms like GitHub	Refuse to use data-related third-party platforms	Host function-similar platforms in local server
Traceability of Analysis Result Quality	Hard to confirm what causes the unexpected result and take too much time	Implement any modules with the confirmation of other participants	<ol style="list-style-type: none"> 1. Utilize automation tools to reduce unnecessary time; 2. Optimize the process;

them to some degree, the way of DevOps enhancement can provide with a more effective solution by not only automation tools, but also process optimization.

C. Long-term Sustainability

Until now, the DevOps community is constantly releasing new open-source automation tools to support DevOps. As mentioned in previous section, implementing a toolchain is the crucial point of D^2Ops functional demand. Given open-source tool's variability and diversity, how to pick tools and how to integrate tools are very difficult to settle. It's necessary to propose a stable approach for guidance.

Toward this objective, a potential solution is to encapsulate business processes that contain several tools as generic components. As a consequence, the toolchain will consist of some generic process components. Obviously, this solution can not only keep process features of toolchain, but also improve the flexibility for future changes.

IV. A REFERENCE PROCESS MODEL FOR D^2Ops

A. Participants Analysis

Recently, a controversial role called *DevOps Engineer* catches people's attention. DevOps engineers are required to possess the ability of development and operation. Sometimes they should also play a role of QA engineer. Consequently, it becomes an emerging development pattern that DevOps engineers need to do everything in development process.

But such pattern is harmful for DevOps[18][19]. For data analytical system, it requires new features about data analysis continuously. Due to the heavy business pressure, DevOps engineers can hardly make overall arrangement. On the contrary, a clear distinction in roles may improve efficiency. Thus the participants of D^2Ops process are suggested to be divided into six roles, and responsible for different business layers.

- *Data Engineer* - They are responsible for data governance and data management.
- *Data Scientist* - They should construct credible data models in different application scenarios.
- *Front-end Software Developer* - Besides providing with interactive user interface, in this case, they are more focused on the visualization of data models.
- *Back-end Software Developer* - They take charge of fulfilling the system skeleton and stable program modules, including the implementation of data model.

- *Integration Engineer* - They ought to cope with operation issues, such as installation, integration and deployment.
- *Quality Assurance Engineer* - They are responsible for discovering latent system issues before release.

B. D^2Ops Process Model

Software development is a complex process that involves with various business activities. In Fig. 1, D^2Ops process model outlines participant's activities and their collaborations. Most of participants have to cope with both of *Dev* and *Ops* works, and several activities are labeled in the figure. Here *Dev* is a generalization concept which represents the works except operations.

The root cause that leads to so intricate activity collaborations is the premised constraints (see Section III). For example, data scientists must apply data models to real data in the isolated production environment. Only this way can really verify correctness. However, the activity probably requires data scientists to do some server operations.

As we emphasized in Section III, it's more beneficial to combine process with automation tools. In order to improve the efficiency of D^2Ops process model, we should make it an integrated automation toolchain.

C. Generic Process Components

According to the sustainability demand of D^2Ops , generic process components are very important for implementing a stable DevOps process model. In the past, several researchers have done similar studies on it. Wettinger et al.[17] thought that DevOps artifacts should be divided into two major classes: Node-centric artifacts and Environment-centric artifacts. Babar et al.[8] thought that DevOps model are composed of production management, development, automated testing, ongoing deployment and operational support.

But Wettinger's solution was proposed from a perspective of deployment artifacts, and didn't consider the whole DevOps process. While although Babar took process into account, he didn't consider concrete businesses of data analytical system. Therefore, these solutions are not so suitable for D^2Ops .

Based on the D^2Ops process model, we designed a series of generic process components. Each component is composed of several automation tools that are substitutable. They are designed to solve the given process demands, and able to be the selection criteria for specific automation tools.

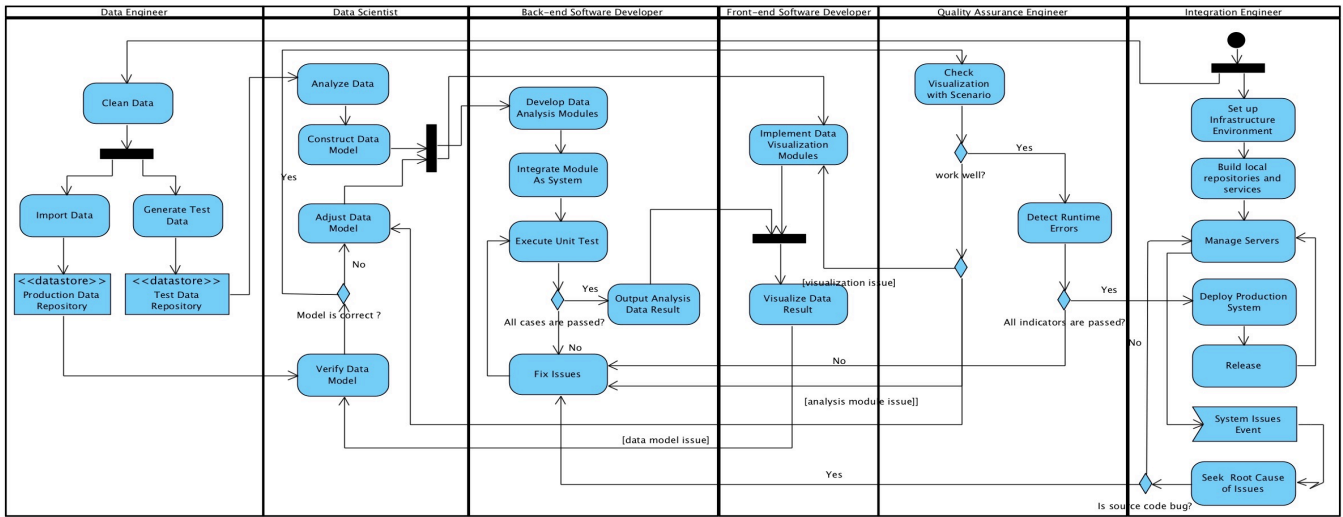


Fig. 1: D^2Ops Process Model

- *Server Management Component* - A component that solves the obstacles of server management. Such as configuration, installation and monitoring.
- *Data Processing Component* - A component that is responsible for data governance. It needs to integrate data from different sources into uniform format and make it meet business requirements. Its work also includes the cleaning of useless data and the storage of processed data.
- *Development Environment Component* - A component that encapsulates development environment. It provides both Dev team and QA team with a unified environment to avoid problems caused by environment difference.
- *Version Control Component* - A component that stores and manages version histories of source codes and artifacts that can recover data models. Its hook mechanism is an important foundation for automatic build.
- *CI Build Component* - A component that takes charge of integrating newcomer source codes. It's a process collection of compiling, building and testing source codes to produce a function module[20].
- *Portability Component* - A component that encapsulates program modules and relevant runtime environment. It's portable and concentrates on solving the troubles caused by isolation restrictions.

V. ESSENTIAL WORKFLOWS OF D^2Ops PROCESS MODEL

In Fig.2, we demonstrate a reference facility for D^2Ops process model on the basis of generic process components. It concentrates on fast feedbacks and portable deployment by automation tools. Parts of available open-source tools are presented in TABLE II. In the perspective of functionality, the facility can be divided into three workflows roughly.

A. Production Development Workflow

In recent years, Docker, an open-source implementation of operating system-level virtualization, is rapidly gaining mind-

share amongst developers[21]. The rise of Docker becomes a strong support on the assumption of *Portability Component*. As a virtual container, it endows users with the capability to package program modules and their runtime environment. Its features, encapsulation and portability, make it suitable for *Development Environment Component* as well.

Due to the isolation situation, the key goal of this workflow is to generate stable *Portability Components* for the production system and data model modules. As is shown in Fig. 2, there should be two repositories in *Version Control Component*. One is for data model artifacts, and another one is for the source codes of production system. When data scientists commit their modification for data models, a feedback from *Version Control Component*, like email, will be sent to developers quickly. Then front-end developers and back-end developers can start to implement the data models and the relevant visualization immediately. Once they also commit source codes by *Version Control Component*, it will trigger hooks set for *CI Build Component*, then run automated testing scripts. Provided that a testing case was failed, *CI Build Component* would send feedbacks to notify both developers and QA engineers. Otherwise, the newcomer source codes will be integrated into lasted source codes and built as *Portability Component*. Afterwards, it's time for QA engineers to do some testing procedures. If everything researches the standard, integration engineers can deploy it to the isolated server. By this workflow, finally, the production

TABLE II. Available tools for generic process components

Component	Open-source automation tools
Server Management Component	Puppet, Chef, Ansible, SaltStack
Data Processing Component	Pentaho Kettle, Talend, KETL
Development Environment Component	Virtual Box, Vagrant, Docker
Version Control Component	<i>Client:</i> Git, Subversion, Mercurial <i>Server:</i> GitLab, Gogs, VisualSVN, Phabricator
CI Build Component	Jenkins, BuildBot, Travis CI
Portability Component	Docker

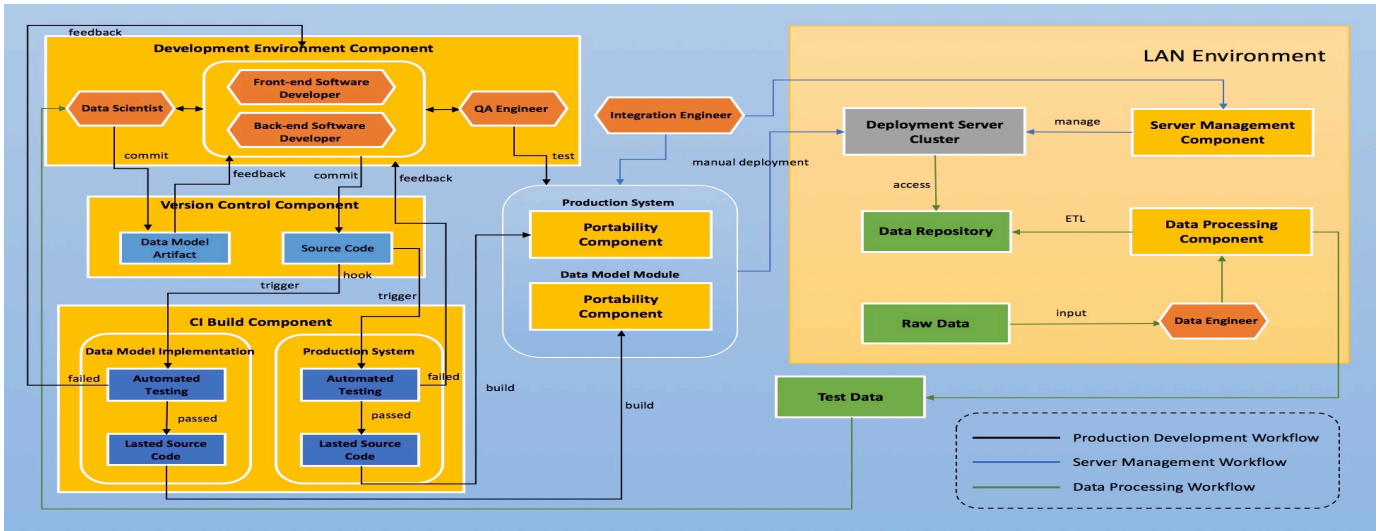


Fig. 2: Reference Facility Based on Generic Process Components

system can run on the server successfully. Meanwhile, data scientists are able to verify their data models.

This workflow is the most critical part in the D^2Ops process model. Its continuous integration way can reduce many unnecessary labor costs and communication costs.

B. Server Management Workflow

As is well known, deployment servers are the most foundational infrastructure. In order to manage server cluster intensively, several open-source automation tools were released. Their core concept is to maintain automation scripts in the master machine, and remotely control minion machines to execute corresponding commands. It's so-called "Infrastructure as Code"[2][22]. By this way, integration engineers can maintain massive servers at the same time. It reduces troubles brought by repetitive work. Moreover, it's also able to monitor server's running status and obtain relevant information constantly.

Nowadays many automated tasks for server management depend heavily on the Internet services. But as illustrated in TABLE I, the constraint, *Network Accessing Control*, disables those services. As a result, integration engineers have to handle all management details. To continue utilizing the feature of those services, a better solution is to publish local services. It requires integration engineers to confirm the needed repository sources on the Internet, and make duplications in local servers.

C. Data Processing Workflow

This workflow connects raw data with production system and data models. It makes the D^2Ops process model become a complete streamline. The workflow mainly focuses on the automation implementation of data governance. When new raw data come into *Data Processing Component*, it will activate this component to clean up dirty data and execute transformation actions automatically. Finally, the processed data will be stored in the *Data Repository*. Meanwhile, the corresponding test data will also be generated. At that time, data scientists can use them to construct data models.

VI. CASE STUDY

As criminal activities are increasingly rampant, traditional ways are unable to satisfy the requirement of solving a crime. Nowadays, utilizing a data analytical system becomes a solid choice for relevant organizations gradually.

We have been appointed to develop an intelligence reconnaissance system called *IAGraph*, which is a kind of data analytical system. Based on the existing criminal-related data, it mainly focuses on the analysis about latent criminal detection. The existing data include criminal records, trip informations, monitoring data and so on. There is no doubt that no matter which of them is leaked, the consequence will be extremely serious.

Under the instruction of D^2Ops solution proposed in this paper, we built this system successfully. Fig. 3 demonstrates our implementation of D^2Ops facility in this case. Each process component is assigned with explicit automation tools, including Vagrant, GIT, Gogs, Jenkins, Docker, Puppet and Pentaho Kettle. Given the possibility of platform migration in the future, all the selected tools are cross-platform.

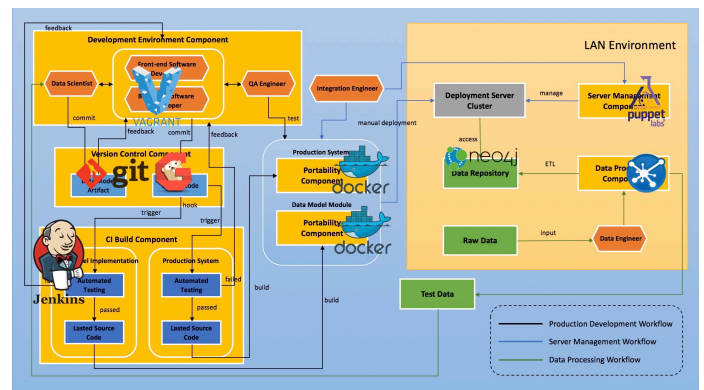


Fig. 3: Reference Facility Implementation

CODE SNIPPET 1: Jenkins Shell Script

```
set -e
CID=$(docker ps | grep "tjsse/iagraph" | awk '{print $1}')
if [ "$CID" != "" ]; then
  docker stop $CID
  docker rmi tjsse/iagraph
fi
rm Dockerfile
wget http://192.168.1.116:3000/DockerfileHub/src/master/IAGraph/Dockerfile
docker build -t="tjsse/iagraph" -f="Dockerfile" .
docker run --name iagraph_run -p 3000:9000 tjsse/iagraph activator run
CCID=$(docker ps | grep "tjsse/iagraph" | awk '{print $1}')
docker commit $CCID ieven/iagraph:deploy
docker save -o IAGraph.tar tjsse/iagraph:deploy
```

CODE SNIPPET 2: Dockerfile Script

```
FROM ingensi/play-framework:latest
MAINTAINER iceplus <zhengjiabin.tj@gmail.com>
ADD ./IAGraph /app
```

Because of the limited space, here we only describe the creation steps of Docker image in detail. The technical key point is to utilize the mechanism of *Git Hook*. In this process, a Jenkins plugin identified by *build-token-root* should be installed necessarily. It endows Jenkins with the ability to build project remotely by HTTP request. After setting an authentication token using this plugin, we configured a related Webhook for git *push* event in Gogos repository. Once developers push their codes to Gogos repository, the Webhook would trigger and send a *POST* request to the specified URL. Then Jenkins would execute the shell script shown in CODE SNIPPET 1 to generate a portable Docker image *IAGraph.tar*. In addition, CODE SNIPPET 2 illustrates the corresponding Dockerfile that defines specific instructions of image creation. Finally, integration engineers could dispatch *IAGraph.tar* to all servers by Puppet, and execute a command “*docker load < IAGraph.tar*” for loading the docker image to Docker platform. Then the deployment work came to an end.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have concentrated on exploring DevOps for data analytical system. After thoroughly analyzing the *D²Ops* demands including various constraints, we proposed a process model for *D²Ops*. Furthermore, we also designed six generic process components to support this model.

Our research achievement takes the constraints of system characteristics as its premise and is able to integrate open-source automation tools as a toolchain effectively. Moreover, it can also be regarded as a reference case. Obviously, the achievement is able to cope with the mentioned challenges when to apply DevOps for data analytical system.

But there are several shortages in our research. Due to the lack of DevOps criteria, we can hardly quantify the effectiveness of *D²Ops*. Moreover, in order to clarify the relation among participant activities, many minor issues were neglected selectively.

For the purpose of improving *D²Ops*, we will be dedicated to solve these shortages in the future. The specific points are presented as follows.

- Conduct abundant relevant investigations and propose a set of evaluation criteria

- Consider more detailed activities
- Carry out *D²Ops* solution in more data analytical systems to collect feedbacks

REFERENCES

- [1] M. Stillwell and J. G. Coutinho, “A devops approach to integration of software components in an eu research project,” in *Proceedings of the 1st International Workshop on Quality-Aware DevOps*. ACM, 2015, pp. 1–6.
- [2] M. Huttermann, *DevOps for developers*. Apress, 2012.
- [3] M. Walls, *Building a DevOps culture*. O’Reilly Media, Inc., 2013.
- [4] S. K. Bang, S. Chung, Y. Choh, and M. Dupuis, “A grounded theory analysis of modern web applications: knowledge, skills, and abilities for devops,” in *Proceedings of the 2nd annual conference on Research in information technology*. ACM, 2013, pp. 61–62.
- [5] L. E. Lwakatare, P. Kuvaja, and M. Oivo, “Dimensions of devops,” in *Agile Processes, in Software Engineering, and Extreme Programming*. Springer, 2015, pp. 212–217.
- [6] N. Ferry, F. Chauvel, H. Song, and A. Solberg, “Continuous deployment of multi-cloud systems,” in *Proceedings of the 1st International Workshop on Quality-Aware DevOps*. ACM, 2015, pp. 27–28.
- [7] D. E. Weeks, “Devops and continuous delivery reference architectures,” <http://www.slideshare.net/SonatypeCorp/nexus-and-continuous-delivery>, <http://www.slideshare.net/SonatypeCorp/devops-and-continuous-delivery-reference-architectures>, accessed March 16th, 2016.
- [8] Z. Babar, A. Lapouchnian, and E. Yu, “Modeling devops deployment choices using process architecture design dimensions,” in *The Practice of Enterprise Modeling*. Springer, 2015, pp. 322–337.
- [9] M. Corp, “Devops and application lifecycle management,” <https://www.visualstudio.com/en-us/features/alm-devops-vs>, accessed March 16th, 2016.
- [10] I. Corp, “Enabling business agility: Devops for the enterprise,” <http://www.ibm.com/ibm/devops/us/en/>, accessed March 16th, 2016.
- [11] A. Corp, “Enabling business agility: Devops for the enterprise,” <https://aws.amazon.com/cn/campaigns/emea-devops/>, accessed March 16th, 2016.
- [12] D. Bruneo, T. Fritz, S. Keidar-Barner, P. Leitner, F. Longo, C. Marquezan, A. Metzger, K. Pohl, A. Puliafito, D. Raz *et al.*, “Cloudwave: Where adaptive cloud management meets devops,” in *Computers and Communication (ISCC), 2014 IEEE Symposium on*. IEEE, 2014, pp. 1–6.
- [13] J. Kim, C. Meirosu, I. Papafili, R. Steinert, S. Sharma, F.-J. Westphal, M. Kind, A. Shukla, F. Németh, and A. Manzalini, “Service provider devops for large scale modern network services,” in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 1391–1397.
- [14] S. Elliot, “Devops and the cost of downtime: Fortune 1000 best practice metrics quantified,” *International Data Corporation (IDC)*, 2014.
- [15] V. P. R. Kumar and V. Babu, “Devops-a review,” *IMAGE AND VIDE PROCESSING*, p. 32, 2012.
- [16] A. M. Christie, *Software process automation: the technology and its adoption*. Springer Science & Business Media, 2012.
- [17] J. Wettinger, U. Breitenbücher, and F. Leymann, “Standards-based devops automation and integration using toasca,” in *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*. IEEE Computer Society, 2014, pp. 59–68.
- [18] I. BJELAJAC, “Devops is a culture, not a job description,” <http://devana.rs/blog/devops-is-a-culture-not-a-job-description/>, accessed March 16th, 2016.
- [19] J. Knupp, “How ‘devops’ is killing the developer,” <https://jeffknupp.com/blog/2014/04/15/how-devops-is-killing-the-developer/>, accessed March 16th, 2016.
- [20] C. A. Cois, J. Yankel, and A. Connell, “Modern devops: Optimizing software development through effective system interactions,” in *Professional Communication Conference (IPCC), 2014 IEEE International*. IEEE, 2014, pp. 1–7.
- [21] J. Fink, “Docker: A software as a service, operating system-level virtualization framework,” *Code4Lib Journal*, vol. 25, 2014.
- [22] D. Spinellis, “Don’t install software by hand,” *Software, IEEE*, vol. 29, no. 4, pp. 86–87, 2012.