DICE enables Quality-Driven DevOps for Big Data

Executive Summary

In July 2014, the EU Commission outlined a new strategy on Big Data, supporting an accelerated transition towards a data-driven EU economy. According to the "Worldwide Big Data Technology and Services, 2012–2015 Forecast" conducted by IDC[1], Big Data services are expected to grow worldwide at an annual growth rate of 40% – about seven times that of the ICT market overall. Another recent study on "Big Data Analytics: An assessment of demand for labour and skills, 2012-2017"[2], conducted by e-skills UK and SAS, predicts that in the UK alone, the number of Big Data specialists will increase by 240% over the next five years. DICE is a response to increasing needs of society to develop Big Data applications.

Today, most organizations face high market pressure, and their supporting ICT departments are struggling to accelerate the delivery of applications and services while preserving production and operations stability. On the one hand, ICT operators lack understanding of the application internals including system architecture and the design decisions behind architectural components. On the other hand, development teams are not aware of operation details including the infrastructure and its limitations and benefits. These issues are even magnified for Big Data systems. For these, recent years have seen the rapid growth of interest for the use of data-intensive technologies, such as Hadoop/MapReduce, NoSQL, and stream processing. These technologies are important in many application domains, from predictive analytics to environmental monitoring, to e-government and smart cities. However, the time to market and the cost of ownership of such applications are considerably high.

We argue that, while typically considered a niche approach for experts, model-driven approaches have the potential to automate several development activities thus helping to **reduce the time to market**. Also, operational data can be exploited using automated tools to give feedback to the development to accelerate the development activities.

In this context, DevOps practices address the common isolation between Development and Operations. The main goal of DevOps is to achieve a better and continuous delivery application lifecycle by eliminating silos and integrating 'Dev' and 'Ops' activities, with people sharing the same goals and working closely with automated tools and services that enables such interactions.

DICE aims at incorporating within the DevOps movement a model-driven approach that facilitates flow of information from Dev to Ops and enables operations monitoring and anomaly detection capabilities to facilitate flow of information from Ops to Dev. Moreover, DICE offers the following tenets in the context of DevOps:

- It enables the data-intensive application design with proper operational annotations that facilitate design time simulation and optimizations. 'Dev' and 'Ops' can work side by side to improve and enhance application design using simulation and optimization tools that offers refactoring of architectural designs.
- It includes analysis tools that exploit monitoring data to detect anomalies and to optimize the configurations of applications, and feedback loop tools that exploit the same data to offer feedback to the architectural design to identify the bottlenecks and refactor architecture.

[1] https://ec.europa.eu/digital-single-market/en/news/worldwide-big-data-technology-and-services-2012-2015-forecast
[2] http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=6243

- DICE also offers tools for continuous integration and automated delivery to facilitate the integration between design-time and runtime.

The current business demand in software industry

Software vendors are nowadays pressured by the market to deliver their product more rapidly and with a higher quality and reliability. Organizations are now challenged to exploit the data in a more intelligent fashion using data-intensive applications that can quickly analyse huge volume of data in a short amount of time. However, Big Data systems are complex systems involving many different frameworks which are entangled together to process the data. How will the software vendors perform at increasingly high levels at the exploding complexity? How will they accelerate time to market without reducing quality of the service?

This increases Big Data application design and operational requirements, and also demands for a common approach in which development and operation teams are able to react in real time throughout the application lifecycle and assure quality and performance needs.

Therefore, Big Data system development introduces a number of technical challenges and requirements for systems developers and operators that will severely impact on sustainability of Business Models, especially for software vendors and SMEs that have limited resources and do not have the strength to influence the market. In particular, the DICE team identifies the following problems in practice:

Lack of automated tools for quality-aware development and continuous delivery of products that has been verified in terms of quality assurance. In other words, there exists many different frameworks that software vendors can adopt in order to develop their data-intensive applications. However, there is no automated mechanism to enable them to verify software quality across such different technological stack in an automated fashion.

Lack of automated tools for quality-driven architecture improvements. In order to develop a data-intensive application, software vendors require to adopt an architectural style in order to integrate different components of the system. During development, the complexity of such data architecture will be increased and there is no automated tool to enable designer to improve the architecture based on performance bottlenecks and reliability issues that have been detected based on monitoring of the system on real platforms.

Software vendors have always found the way to adapt to new conditions to help organizations adopt and reach their goals. During the last years, the DevOps practices has gained significant momentum [1]. Organizations from across industries have recognized the need to close the gap between development and operations if they want to remain innovative and responsive to today's growing business demands. This new paradigm focuses on a new way of doing things, an approach that tries to improve the collaboration and communication of developers and operators while automating the

process of software delivery and infrastructure changes. Therefore, a new ecosystem is needed to facilitate such automated processes by offering the following capabilities:

Business Needs and Technical Quality of Service: All types of organizations need to move fast, and need to align IT assets to their needs (create new assets or adapt existing ones). Therefore, ICT departments need to assure that their ICT platforms and infrastructures are flexible enough to adopt business changes, regardless of the type of IT infrastructures used to create those data-intensive applications, whether they are built using open-source software or data services offered by public cloud providers. In other words, they should avoid technological lock-in as much as possible.

Architecture Refactoring: efficiency, reliability and safety of applications should be monitored in testing and production environments, with metrics and data that feedback directly and quickly to development teams for faster testing, improvement and adjustment to meet service level goals.

Application Monitoring and Anomaly Detection: Access to live data gathered by monitoring engines should also provide performance and reliability data for observed applications in production to gauge the need for identifying outliers and detecting any anomaly that may harm continuous business processes that are dependent to such data-intensive applications. Moreover, such monitoring data should assist application architects and developers in building toward an optimized target infrastructure.

Efficiency and optimal configuration: Big Data applications typically spend expensive resources that are offered to companies via public clouds. Therefore, it is of utmost importance to optimize such applications in order to demand less resources and produced more output. Therefore, organizations require to have automated tools to optimally configure such application without the need to hire experts to optimize their Big Data applications.

Why a DevOps solution is required to address Big Data software development challenges?

DICE helps both developers and operators. The tools that DICE provides help lower existing barriers between Development and Operations Teams and help embrace DevOps practices to change and improve the way data-intensive software is created and operated. This means that DICE ecosystem can help software vendors of any size to Build and Run Big Data Applications by business and technical needs and quality requirements.

The DICE solution can be understood as a set of supporting tools that underpins DevOps strategies since it helps organizations empower development and operation teams with the following capabilities:

- DICE takes both Business and Technical perspectives into account when designing Big Data application architectures and their operational requirements. This ensures that the resulting

applications comply with business and functional quality levels by exploiting automated quality assessment that usually developers manually perform.

- DICE provides a framework that helps developers to exploit the knowledge gained from operational data on real cluster that the data-intensive applications are deployed to improve the performance of the system by optimizing the configuration as well as refactoring the architecture
- DICE Monitoring capabilities give DevOps teams an overview of operational data that will help design and operate applications in a more optimized way.
- DICE allows the possibility of providing performance and model driven approaches to improve current DevOps practices such as continuous delivery for the data-intensive application pipeline.
- DICE helps blurring the line between 'Dev' and 'Ops' teams, by providing them with new tools that support the whole life-cycle phases of a data-intensive software, putting them in position to better satisfy business needs.

Therefore, DICE embraces the idea of DevOps movement where development and operations teams are seeking new ways to work together and enhance overall system performance along the full application life-cycle (plan, design, build, deploy, test, operate and maintain) to comply with business and technical requirements.

The following sections of this white paper present DICE (Developing Data-Intensive Cloud Applications with Iterative Quality Enhancements) solution and a real use case scenario to explain how it can help organizations adopt modern workflows with the right tools.

DICE technical solution and its unique advantages

A high-level overview of the DICE framework is shown in Figure 1. This framework includes the following components:

- **DICE IDE**: an integrated development environment to accelerate coding, design and application prototyping based on the DICE Profile for UML and the DICE Methodology;
- **Quality analysis tools**: a set of tools for quality analysis during the early-stage of application design via simulation, verification and optimization methods;
- **Feedback and Iterative Enhancement tools**: a monitoring platform tailored to Big Data technologies and coupled with tools that will continuously detect quality anomalies that affect the application design and explain how the data-intensive application utilizes resources;
- **Continuous Delivery and Testing tools**: a set of tools and methods supporting delivery on private and public clouds via a TOSCA-compliant[3] deployment tool, optimal application configuration, continuous integration, and quality testing.

[3] https://www.oasis-open.org/committees/tosca/

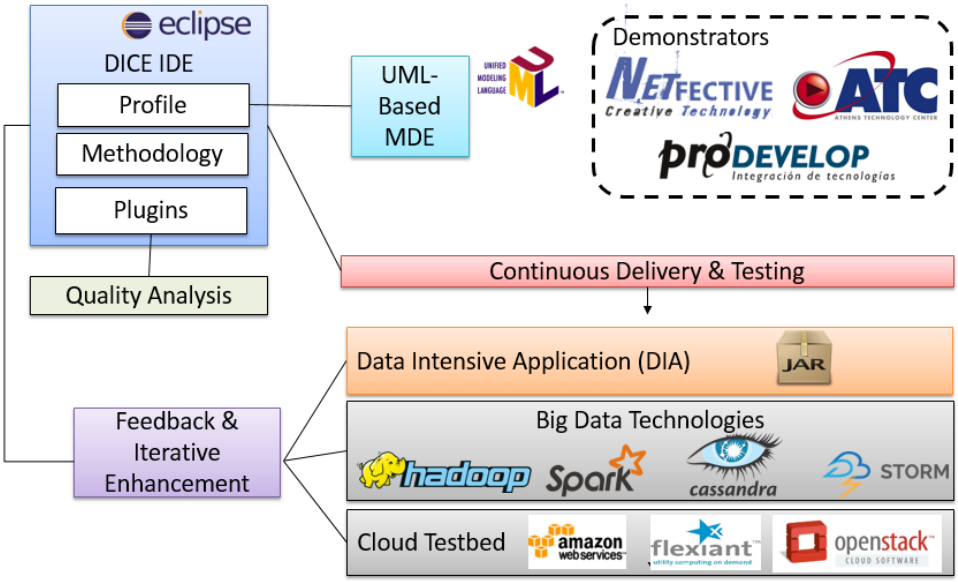

Figure 1 DICE Framework

The above architectural components are further break down in several tools, each one belonging to one of two categories:

- **Development tools,** which are primarily centered on the development stage of the data-intensive application. The IDE implements the DICE quality-driven methodology that guides the developer through the different stage of refinement of design models up to implementing the initial prototypes of its application. The IDE supports multiple quality analyses offered by the verification, simulation and optimization tools.
- **Runtime tools,** which collect data during the application quality testing to characterize the efficiency, reliability and correctness of the components. This data is consolidated in the DICE monitoring platform and used to semi-automatically detect anomalies, optimize the system configuration, and enhance the design.

The purpose of the tools within each group of tools is explained in details in the following sub sections.

Development tools

The central element of the DICE architecture is the **Integrated Development Environment** (**IDE**), where the developer specifies the data-intensive application using a model-driven engineering approach. To support this activity, the DICE Eclipse-based IDE embeds the **DICE UML profile** which provides the stereotypes and taggued values needed for the specification of data-intensive applications in UML.

Following an Model-Driven Architecture (MDA) approach, models are defined by the user in a top-down fashion, stemming from platform-independent specifications of components and architecture (**DICE Platform Independent Models**), through assignment of specific technologies to implement such specifications (**DICE Technology Specific Models**), and finally to the mapping of the application components into a concrete TOSCA-compliant deployment specification (**DICE Deployment Specific Models**). Such UML models can be related by DICE model-to-model transformations that are

automatically executed within the IDE, to reduce the amount of manual work required from the user. For example, initial DTSM and DDSM models can be generated from the DPIM models.

Throughout the application design, the DICE IDE offers the possibility to automatically translate certain DICE models into *formal models* for assessment of quality properties (efficiency, costs, safety/correctness, etc.). Each analysis requires to run dedicated tools that reside outside the IDE environment, in order to obtain prediction metrics. The **simulation**, **optimization** and **verification plugins** take care of translating models in-between IDE and these external tools. They also collect via REST APIs the outputs of these tools that are shown to the user inside the IDE. The interface of these plugins assumes the user to be unskilled in the usage of the formal models. Furthermore, the quality properties are defined in terms of constraints using appropriate language constructs.

As the developer progressively refines the application model and the application code, s/he is going to periodically commit them to a **repository, i.e., a version control system (vcs)**. In DICE we will ensure that every commit increases the **version number** of the application, which is a unique identifier used across tools to keep synchronized models and code. The repository acts as a shared source of models and code across different versions for all the DICE tools that need to access them. The repository will reside externally to the IDE and will be accessed through appropriate tools (e.g. SVN, GIT, etc.).

Runtime tools

After the initial prototyping of the application, the developer will request to deploy the current prototype. After an automatic commit of all models and code to the external repository, the **continuous deployment tool** will retrieve a copy of both of them from the repository, build the application, and internally store the outputs and their associated artifacts. The delivery tool will then initialize the deployment environment (if not already created), consisting of VMs and software stack, and deploy (or update the existing deployment of) the application. The delivery operation also connects the DICE **monitoring platform** to the deployed application. The monitoring platform will be started and stopped by REST APIs and will acquire a pre-defined set of metrics that will be continuously stored in a performance data repository.

The **anomaly detection** and **trace checking tools** will also feature an IDE plugin and will be able to query the monitoring platform for relevant metrics and use them to generate analyses concerning anomaly in performance, reliability or operational behavior of the application at a given release version. The anomaly detection tool will reason on the base of black-box and machine-learning models constructed from the monitoring data. Conversely, the trace checking tools are going to analyze the correctness of traces. These analyses will be manually invoked by the user from the IDE plugin. Similar to these, the **enhancement tool** will automatically annotate the DICE models stored in the repository with statistics on the inferred and recorded monitoring data, in order to help the user to inspect the root-causes of performance or reliability anomalies.

The **quality testing tools** will support the generation of test workloads to the application. Such workloads are those that will be used to assess the quality of prototypes. Similarly, the **fault injection tool** will generate faults and malicious interferences that can help verifying the application resilience. Both tools integrate a heterogeneous set of actuators and can be run manually by an operator through

a command-line interface and configuration files. They will also be exploited by the **configuration optimization** tool to generate an experimental plan automatically given a time budget. The output of this tool is to confirm the optimal configuration of the deployment for an application in its final stages before being pushed to production. Compared to the optimization plugin, configuration optimization will also deal with fine-grained systems parameters (e.g. buffer size, block size, JVM configuration, etc.), which are difficult to model in design-time exploration. Moreover, configuration optimization is black-box and solely measurement-driven, whereas design space exploration is primarily model-driven.

The DICE framework is being evaluated using three industrial demonstrators in the areas of news and media (provided by ATC), maritime operations (provided by Prodevelop), and tax fraud detection application (provided by Netfective Technologies):

- **ATC Demonstrator (News&Media)**: ATC provides a use case scenario on NewsAsset, a commercial data-intensive application positioned in the media domain that efficiently gathers news media items (text, video, images, etc.) from heterogeneous nodes, utilize services to aggregate them to relax redundancy and compose services to associate relevant data coming from different sources. Even though the current platform reached the envisioned level of success with respect to its original objectives, it still lacks of efficiency and performance when operated in real life scenarios. DICE is envisioned to tackle these quality issues.
- **NETF Demonstrator (e-Government)**: Netfective Technology, as a partner in DICE, is providing a functional and technical demonstrator aiming to illustrate the added value of Big Data in e-government applications especially for tax fraud detection. Tax evasion and frauds represent a huge problem for governments, causing them a big loss of money each year. The EU estimates the sums are being lost due to tax evasion and avoidance of the order of € 1 trillion [3]. This demonstrator aims to build a sample of a template model of "fraudulent conduct" from the automatic operations on existing tax data files such as business creation or bank accounts abroad which represents millions of data points. Some examples to be detected by the fraud detection demonstrator include identifying taxpayers who are registered in different local administrative areas in order to collect fraudulently social aids or the fictitious relocation of the taxpayer who improperly claims to be domiciled abroad in order to not pay tax on income or on wealth.
- **PRO Demonstrator (Maritime Operations):** The geo-fencing demonstrator scenario was defined based on Posidonia Operations product. This use case scenario involves a data-intensive application positioned in the maritime domain that efficiently gathers real-time positions of vessels (AIS frames) from AIS emitters on ships, utilizes services to parse and merge them to avoid redundancy and extract relevant data coming from different AIS sources. Although Posidonia Operations has been successfully deployed in various Port Authorities, it still lacks cloud-based support, efficiency and performance when deployed in different environments, and quality analysis support. DICE is envisioned to tackle these quality issues by offering tools to revise/reconfigure the current architecture with respect to cloud processing, Big Data technologies and quality metrics, integrate data-intensive legacy components and manage complexity when temporal peaks of high compute demand occur.

Conclusions

This white paper presents DICE project ongoing research on implementing a framework and IDE for developing and deploying data-intensive applications to improve the way development and operations team work and align their efforts to be more agile, efficient and provide the means to comply with business needs to survive in today's demanding market conditions. DICE ecosystem is intended to facilitate organizations embrace performance-driven DevOps strategies in the context of Big Data applications.

DICE intends to help software vendors embrace DevOps with a proper tool chain engineered to provide automated assistance throughout the full application lifecycle both from Development to Operation as well as the way back from Operation to Development. It starts enabling Development teams translating business and technical requirements early. These requirements will be translated into Operations, and thanks to the monitoring engine, feedback will be provided back to Dev teams and automated decisions will assure the correct execution and adaptation of data-intensive systems.

In order to support the development of high-quality data-intensive applications, DICE delivers a set of core innovations that will benefit software vendors:

- **Tackling skill shortage and steep learning curves in quality-driven development of data-intensive software** through open source tools, models, methods and methodologies that are aware of data and can model Big Data technologies (e.g., Hadoop, NoSQL). Automation is a distinctive concern in DICE for integrating quality engineering in the development process transparently. The software engineer does not need to pre-possess specific knowledge on quality assessment and enhancement.
- **Shortening the time to market** for data-intensive applications that meet quality requirements, thus reducing costs for software vendors while at the same time increasing value for end-users.
- **Decreasing costs to develop and operate** data-intensive cloud applications, by defining algorithms and quality reasoning techniques to select optimal architectures, especially in the early development stages, and taking into account SLAs.
- **Reducing the number and severity of quality-related incidents and failures** by leveraging DevOps-inspired methods and traditional reliability and safety assessment to iteratively learn application runtime behaviour. As a consequence, the architecture is refactored automatically, so to address the root causes of the incidents.
- **Optimized operation** by leveraging experimental data that has been collected intelligently in order to optimize the data-intensive software with the best configuration.

Acknowledgement

The DICE project (February 2015-January 2018) has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644869

References

[1] Andreas Brunnert, et al., "Performance-oriented DevOps: A Research Agenda", SPEC Research Group --- DevOps Performance Working Group, Standard Performance Evaluation Corporation (SPEC), SPEC-RG-2015-01 (2015).

[2] G. Casale, D. Ardagna, M. Artac, *et al*. DICE: Quality-Driven Development of Data-Intensive Cloud Applications. Proceedings of the 7th International Workshop on Modeling in Software Engineering (MiSE 2015).

[3] Tax Fraud evasion: a Huge Problem, last accessed 21 April 2016: http://ec.europa.eu/taxation_customs/taxation/tax_fraud_evasion/a_huge_problem/index_en.htm

DICE partners

ATC:	Athens Technology Centre
FLEXI:	Flexiant Limited
IEAT:	Institutul e-Austria Timisoara
IMP:	Imperial College of Science, Technology & Medicine
NETF:	Netfective Technology SA
PMI:	Politecnico di Milano
PRO:	Prodevelop SL
XLAB:	XLAB razvoj programske opreme in svetovanje d.o.o.
ZAR:	Unversidad de Zaragoza

Contact information and additional material on DICE can be found at: http://www.dice-h2020.eu