



DataBench

Evidence Based Big Data Benchmarking to Improve Business Performance

D5.3 Assessment of Technical Usability, Relevance, Scale and Complexity

Abstract

This document proposes an evaluation and monitoring approach for the DataBench framework based on the current ISO/IEC standards for system and software quality. The developed methodology adapts multiple quality factors implemented using technical metrics in order to assess the DataBench framework capabilities from different user perspectives. Furthermore, the approach is integrated in the existing DataBench workflow and modular architecture, and can be implemented using the technology frameworks applied in the Alpha release (deliverable D3.2) of the DataBench Toolbox. The resulting functionality can be seen as a set of interactive evaluation reports realized in the form of web dashboards depicting multiple technical metrics for the different user-roles in the DataBench framework.

This document is the second deliverable in WP5 after D5.1 reporting the results from the activities in Task 5.2 related to the technical usability, relevance, scale, complexity and other quality metrics of the DataBench Framework.



Deliverable D5.3	Assessment of technical usability, relevance, scale and complexity
Work package	WP5
Task	5.2
Due date	30/06/2019
Submission date	27/06/2019
Deliverable lead	GUF
Version	1.0
Authors	GUF (Todor Ivanov, Timo Eichhorn) JSI (Marko Grobelnik, Inna Novalija) ATOS (Tomás Pariente, Iván Martínez and Ricardo Ruiz)
Reviewers	SINTEF (Arne Berre) ATOS (Tomás Pariente, Iván Martínez and Ricardo Ruiz)

Keywords

Benchmarking, big data, big data technologies, architecture, business performance, performance metrics, toolbox, use cases

Disclaimer

This document reflects the authors view only. The European Commission is not responsible for any use that may be made of the information this document contains.

Copyright Notice

Copyright belongs to the authors of this document. Use of any materials from this document should be referenced and is at the user's own risk.

Contents

Executive Summary	5
1. Introduction	6
2. Evaluation Objectives	8
3. DataBench Evaluation Methodology	11
3.1 Factors and Features	11
3.2 Metrics	12
3.3 Evaluation Process	14
4. DataBench Evaluation Implementation	18
4.1 DataBench Methodology Workflow	18
4.2 DataBench Framework Architecture Modules.....	20
4.3 DataBench Framework Technology Implementation	22
5. DataBench Knowledge Graph Monitoring.....	26
6. Conclusion.....	28
References	29

Table of Figures

Figure 1. Functional view of the DataBench ecosystem and WPs	6
Figure 2. Relationships between System and Software Product Quality Requirements and Evaluations using the ISO/IEC Standards. Division of the different families of Standards. [5]	9
Figure 3. DataBench Evaluation Process	14
Figure 4. DataBench Methodological Framework.....	18
Figure 5. Functional Overview of the DataBench Framework Architecture [13]	21
Figure 6. Overview of the Three Metrics Dashboards.....	22
Figure 7. Overview of the technical implementation of the Alpha version [14]	23
Figure 8. Example for Platform Metrics Dashboard.....	25

Table of Tables

Table 1. DataBench Metrics mapped to Quality Factors.....	12
Table 2. Example Metrics resulting from the DataBench Evaluation Process	15

Executive Summary

The goal of Task 5.2 is to assess the technical capabilities of the DataBench framework by developing a systematic methodology and define accurate metrics that can evaluate the DataBench framework with respect to usage and utilization. This document describes our initial approach to develop such a methodology and the necessary technical changes of the framework in order to implement the metrics.

The document starts with brief introduction of the evaluation objectives and the quality characteristics that need to be guaranteed by today's software systems. Then, we present our evaluation and monitoring approach based on the current ISO/IEC standards for system and software quality. The developed methodology adapts multiple quality factors implemented using technical metrics in order to assess the DataBench framework capabilities from different user perspectives. Using table structure, we present the mappings between the quality factors and the metrics as well as give examples to motivate the need of these metrics. Next, we describe how our approach can be integrated in the existing DataBench workflow and modular architecture by implementing the defined metrics in the technology frameworks already used in the Alpha release (deliverable D3.2) of the DataBench Toolbox. The resulting functionality presents a set of interactive evaluation reports realized in the form of web dashboards. We defined three types of dashboards (Platform, User (Profile) and Administrator) with multiple technical metrics to represent accurately the different user-roles in the DataBench framework.

The evaluation methodology and metrics presented in this document will be implemented in the next releases of the DataBench Toolbox.

1. Introduction

This document presents the concept and implementation of technical evaluation methodology as part of the DataBench Framework. The theoretical approach of the evaluation methodology is based on the latest system and software quality requirement and evaluation standards defined by ISO/IEC organization described in Section 2. Our approach adapts the quality characteristics and defines appropriate metrics, which will be implemented in the DataBench framework and will offer monitoring and evaluation capabilities to the different roles of users (business, technical or administrators). The focus of this evaluation is on the technical aspects of the framework and does not cover the validation of the mapping between the technical and business KPIs and metrics. The technical metrics defined as part of the evaluation methodology cover the current state of the DataBench framework and new metrics will be added as the framework development progresses.

The present document reports the findings in Task 5.2 which is part of Work Package 5 (WP5) focusing on the validation, evaluation and assessment of the sustainability of the toolbox. As depicted in Figure 1, the report is tightly coupled with the development of the DataBench Toolbox presented in deliverables D3.1 (DataBench Architecture) and in D3.2 (DataBench Toolbox Alpha version) from WP3. However, as the toolbox development and evaluation are going in parallel the methodology and metrics presented in this document will be implemented in the future releases of the toolbox.

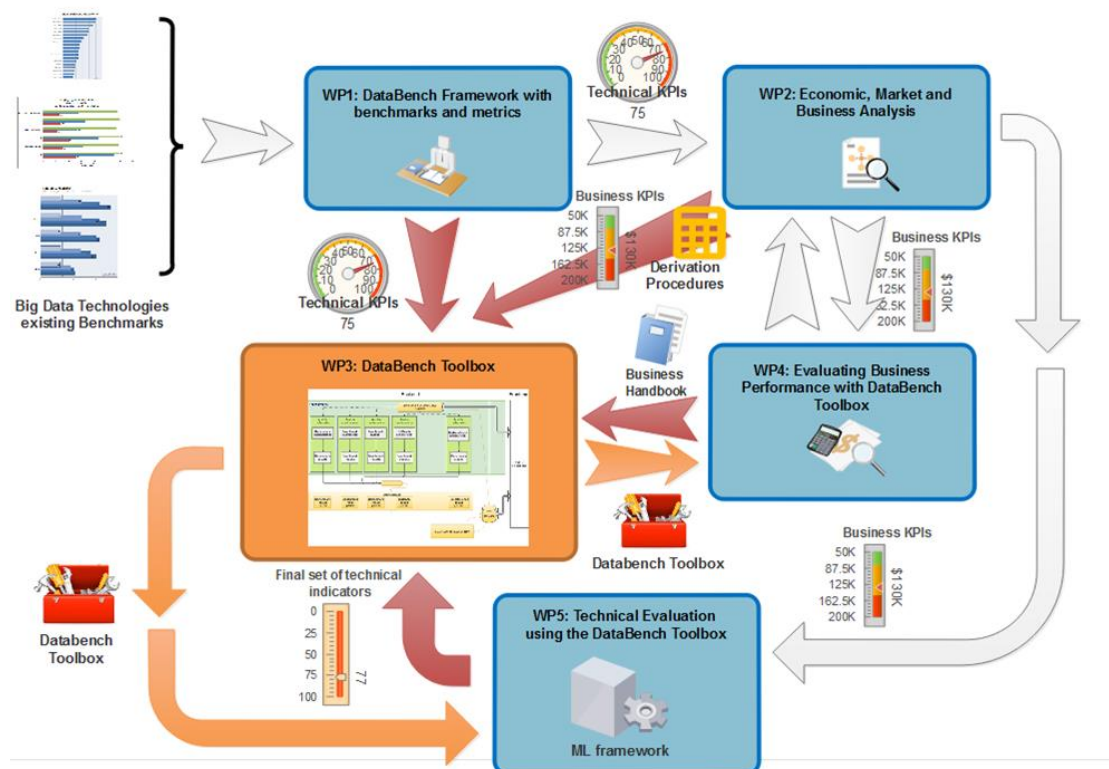


Figure 1. Functional view of the DataBench ecosystem and WPs

The document is structured as follow:

- Section 1 provides the introduction to the deliverable.

- Section 2 describes the objectives and motivation to develop an evaluation methodology and process.
- Section 3 presents in detail the DataBench technical evaluation methodology.
- Section 4 describes the integration and implementation of the evaluation metrics into the DataBench framework.
- Section 5 presents some insights on the monitoring of the DataBench Knowledge Graph.
- Finally, Section 6 provides the conclusions of the document as well as outlines to the future work.

2. Evaluation Objectives

The main objective of Deliverable D5.3 is to assess the **technical usability, relevance, scale** and **complexity** of the DataBench framework with respect to usage and utilization. To understand better how to achieve this goal, we first clarify the terminology and identify the key metrics that need to be implemented and measured in order to perform the overall framework evaluation.

The term **usability** refers to a set of multiple concepts, such as execution time, performance, user satisfaction and ease of learning ("learnability"), taken together and has not been defined homogeneously, either by the researchers or by the standardization bodies [1]. Historically, there were multiple ISO standards defining usability and focusing on product-oriented standards (ISO 9126, 2001; ISO 14598, 2001) or process-oriented standards (ISO 9241, 1992/2001; ISO 13407, 1999).

The term **relevance** refers to the degree with which the implemented software product addresses the user expectations and satisfies the system and user requirements. A higher degree of relevance guarantees the tool acceptance and use among the users. This is further supported by the term **scale**, which in this context represents the necessary measures in order to objectively assess the software tool with respect to its usage and feature coverage. Other necessary evaluation aspects are the tool **complexity** and **extensibility**. The complexity measure represents both the complexity of internal architecture as well as the complexity of usage and the degree of specific knowledge that the user needs to have in advance. The **extensibility** feature represents the ability to add new features and extend the functionality of the software tool.

In addition to the above listed evaluation characteristics there are more quality metrics such as **performance efficiency, compatibility, reliability, security, maintainability, portability** and **supportability** [2], which are also very relevant when implementing and evaluating the quality of a software tool. The performance efficiency focuses on measuring the execution times, resource utilization, data usage and similar other resources utilized by the software tool. The compatibility characteristic of the evaluated tool with respect to other tools, standards and system environments together with the tool reliability, maintainability and portability are key requirements for assuring the high quality of the developed software.

As all of the above listed characteristics became more and more relevant in the software development process, the existing ISO standards on usability have been replaced with new one. The ISO/IEC organization published new series of standards about System and software Quality Requirement and Evaluation ("SQuaRE", also "25000 series standards") that address the quality and evaluation of the entire software lifecycle [3]. This series of standards specifies quality model, measurement, requirement, and evaluation process applied to system and software quality assessment, and it can help to develop and capture systems and software products using quality requirements and evaluations [4]. The relations between these standards and the different families of standards are depicted on Figure 2. The five major division of SQuaRE standards are [4]:

- **ISO/IEC 2500n - Quality Management Division.** The International Standards that form this division define all common models, terms and definitions further referred to by all other International Standards from the SQuaRE series. The division also provides requirements and guidance for a supporting function that is responsible for the

management of the requirements, specification and evaluation of software product quality.

- **ISO/IEC 2501n - Quality Model Division.** The International Standards that form this division present detailed quality models for computer systems and software products, quality in use, and data. Practical guidance on the use of the quality models is also provided.
- **ISO/IEC 2502n - Quality Measurement Division.** The International Standards that form this division include a software product quality measurement reference model, mathematical definitions of quality measures, and practical guidance for their application. Examples are given of internal and external measures for software quality, and measures for quality in use. Quality Measure Elements (QME) forming foundations for these measures are defined and presented.
- **ISO/IEC 2503n - Quality Requirements Division.** The International Standards that form this division help specify quality requirements, based on quality models and quality measures. These quality requirements can be used in the process of quality requirements elicitation for a software product to be developed or as input for an evaluation process.
- **ISO/IEC 2504n - Quality Evaluation Division.** The International Standards that form this division provide requirements, recommendations and guidelines for software product evaluation, whether performed by evaluators, acquirers or developers. The support for documenting a measure as an Evaluation Module is also present.

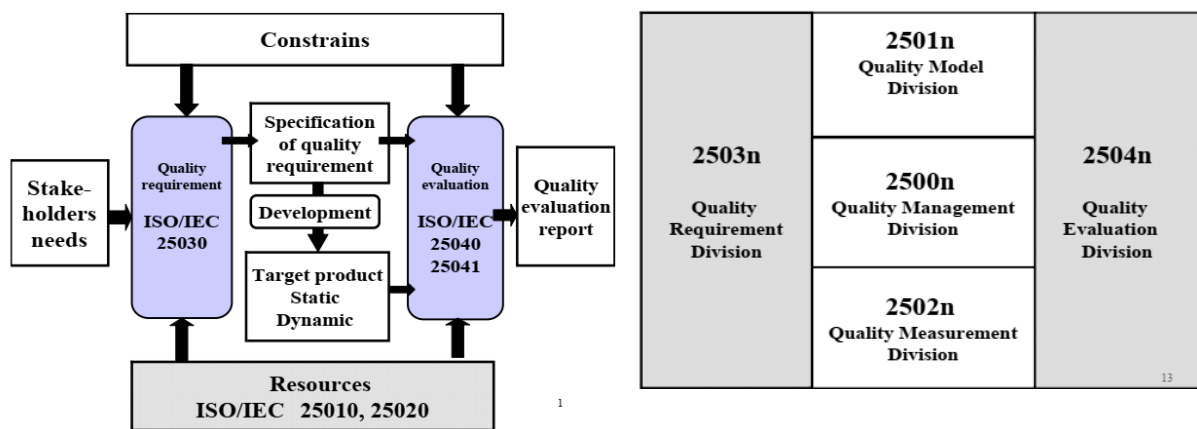


Figure 2. Relationships between System and Software Product Quality Requirements and Evaluations using the ISO/IEC Standards. Division of the different families of Standards. [5]

Recently many related studies and projects have adopted the system and software quality requirements and evaluations defined by the above ISO standards. Hussain and Mkpjojiogu [6] applied the ISO/IEC 25010 standard to evaluate the quality of e-Ebola Awareness System web-portal, while Ouhbi et al. [7] applied the standard to create a software product quality model to present a checklist with which to calculate the influence of the mobile personal health record (mPHR) requirements on software product quality. Applying the above ISO standards, the master thesis of Maria Isabel Ortega Canalejo [8] proposes a quality model for evaluation of Big Data Database technologies. Similarly, Nakai et al. [9] and Tsuda et al. [10] propose a SQuARE-based software quality evaluation framework using product and

quality metrics defined by the SQuaRE series and validate the framework by applying it on commercial software products. Another relevant approach to ensure the quality of service is defined in ISO/IEC 13236:1998 “Quality of Service” [11].

Following the same way of thinking in our approach we are applying the ISO/IEC standards on system and software quality to define and implement metrics into the DataBench framework. Then based on the metrics we can evaluate the platform with respect to different features and categories defined in the ISO SQuaRE standards. In the subsequent sections we explain in detail our methodology and how we plan to implement it as part of the DataBench framework.

3. DataBench Evaluation Methodology

The DataBench Evaluation Methodology covers the different core elements of the described ISO/IEC standardizations [4]. In the following section, these elements are adapted in the evaluation methodology of the DataBench project. The different factors and features are briefly described, and the metrics measuring these are also defined. Finally, the entire evaluation process is explained.

3.1 Factors and Features

First, we adapted the quality factor definitions from the ISO SQuaRE standards [4] and described their relation to the DataBench framework.

Quality in use characteristics

- **Effectiveness**, is the accuracy and completeness with which users achieve specified goals. It is important to track the successful completion of the tasks (e.g. benchmark execution) started by the DataBench users.
- **Efficiency**, resources expended in relation to the accuracy and completeness with which users achieve goals. The necessary time to run a desired task should be minimized.
- **Satisfaction**, degree to which user needs are satisfied when a product or system is used in a specific context of use. Sub-characteristics are usefulness, trust, pleasure, and comfort.
- **Freedom of risk**, degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment. To minimize the economic risk only reviewed benchmarks and their execution files should be integrated into DataBench.
- **Context coverage**, degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in both specified contexts of use and in contexts beyond those initially explicitly identified. The DataBench framework is developed as a general purpose tool which can be applied in both on-premise environments and in multiple user scenarios.

Product quality characteristics

- **Functional suitability**, degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions. The documented benchmarks should be runnable and there should be guidance which specific benchmark the user has to run to meet his business needs.
- **Performance efficiency**, performance relative to the amount of resources used under stated conditions. Sub-characteristics are time behavior, resource utilization, and capacity. The DataBench framework should be always available to users to search for suitable benchmarks and deploy the DataBench Toolbox both in cloud and on-premise.
- **Compatibility**, degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment. The benchmarks should be executable on different infrastructures (on-premise / cloud) and hardware.

- **Usability**, degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. The user-friendly interface including navigation help and the high number of integrated benchmarks are the basis for the DataBench framework usability.
- **Reliability**, degree to which a system, product or component performs specified functions under specified conditions for a specified period of time. This also includes fault tolerance and recoverability. In DataBench framework, the benchmark automation scripts will guarantee the reliability of installation and execution. Any runtime errors and exceptions will be reported by the DataBench Toolbox.
- **Security**, degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. As the benchmarks are run on the infrastructure of the user, special attention should be paid on the commands executed to prepare the benchmarks as well as the credentials stored in the DataBench Toolbox.
- **Maintainability**, degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. Important sub-characteristics for the DataBench project are: Reusability, modifiability, and testability. The DataBench tool will be open source which will allow the users to reuse, extend, test and modify the available functionalities.
- **Portability**, degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. Adaptability and installability are the main sub-characteristics for the DataBench Toolbox.

3.2 Metrics

To measure the above listed quality characteristics it is necessary to define appropriate metrics that collect, evaluate and report the precise data. Table 1 describes the DataBench metrics with definition and mapping to the relevant quality factors.

Table 1. DataBench Metrics mapped to Quality Factors

Metric	Description	Mapping to factor
<i>Rate of successful tasks:</i> $\frac{\text{Number of successful benchmark runs}}{\text{Total number of benchmark runs}}$	Percentage of successful benchmark runs. A drill-down to a specific task or user should be implemented.	Effectiveness
<i>Average execution time of successful tasks:</i> $\frac{\sum_{j=1}^R \sum_{i=1}^N n_{ij} t_{ij}}{\sum_{j=1}^R \sum_{i=1}^N t_{ij}}$	N = The total number of tasks (goals) R = The number of users n _{ij} = The result of task i by user j; if the user successfully completes the task, then N _{ij} = 1, if not, then N _{ij} = 0	Efficiency

	t_{ij} = The time spent by user j to complete task i. If the task is not successfully completed, then time is measured until the moment the user quits the task. A drill-down into specific users and tasks would provide some additional information.	
<i>Single Ease Question</i>	After a task execution the user should be asked how difficult it was to run the task. Typically, the user can choose between 7 grades from very easy to very difficult.	Satisfaction
<i>Resource utilisation of hosting platform</i>	Are the used resources for the DataBench framework under- or overprovisioned.	Performance efficiency
<i>Number of platform environments</i>	Number of different platform environments on which were executed DataBench benchmarks.	Compatibility, Portability
<i>$\frac{\text{Number of benchmarks executed}}{\text{Number of users}}$</i>	Average of executed benchmarks per user.	Usability
<i>Number of benchmark searches</i>	Number of searches for benchmarks (additionally separated by user and benchmark).	Usability
<i>Number of submitted benchmark results</i>	Number of submitted benchmarks results (additionally separated by user and benchmark).	Usability
<i>Number of use case scenarios</i>	Number of use case scenarios implemented and linked to a benchmark.	Usability
<i>Number of (active) users</i>	Number of users which performed at least one task during the last month. The users should be distinguished to admin, technical users, business users, and benchmark providers.	Satisfaction, Usability, Functional suitability
<i>Uptime of DataBench Toolbox</i>	Reachability of DataBench Toolbox for users.	Reliability
<i>Number of implemented benchmarks</i>	Total number of executable benchmarks. Number of benchmarks added to the Toolbox during <i>delta of time</i> .	Maintainability

<i>Average time spent to add a new benchmark</i>	Time from register until the first successful execution of a benchmark within the DataBench Toolbox.	Maintainability
--	--	-----------------

3.3 Evaluation Process

The applied evaluation process is adapted from Zhao et al. [3] and depicted in Figure 3.

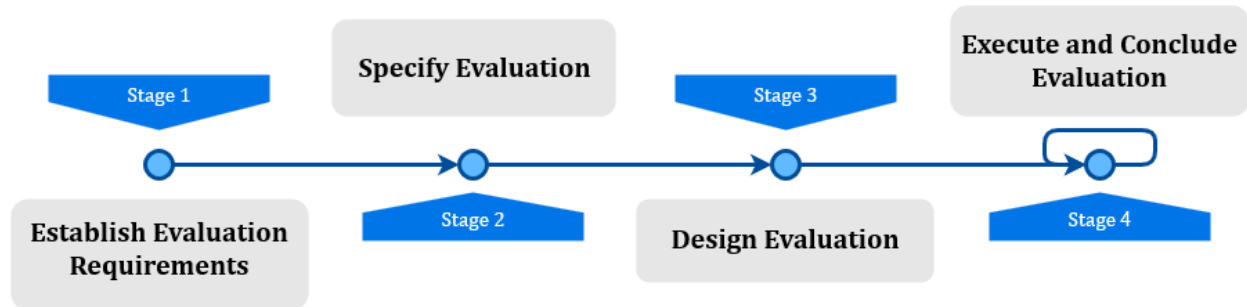


Figure 3. DataBench Evaluation Process

Below we briefly describe the different stages consisting of the following tasks:

Stage 1 - Establish Evaluation Requirements

The first task in this activity is to establish the evaluation purpose for the future activities. The software quality requirements have to be obtained and the product parts which have to be included into the evaluation need to be identified.

Stage 2 - Specify Evaluation

The purpose of this activity is to specify the evaluation modules of the quality measures. The outputs of the previous activity could be the inputs of this activity. The quality measures selected should cover all the evaluation requirements, and also system/software quality characteristics (and/or sub-characteristics) should be measured rigorously.

Stage 3 - Design Evaluation

This activity is to schedule the evaluation activities with the considerations of budget, methods, tools, adopted standards, personnel, etc. The high level evaluation plan defined in the early stage should be revised and adjusted as the evaluation activities evolve.

Stage 4 - Execute and Conclude Evaluation

The evaluation execution is the application of the first three activities, which is to produce the results of software product quality measures and evaluation. Firstly, the quality measurement is performed with the selected quality measures. This generates the results that describe the degree to which the system/software product meets the requirements.

To conclude the evaluation process a joint review of the evaluation results and disposition of evaluation data should be performed. It is essential to note that in the review and feedback task, not only the evaluation results, but also the validity of evaluation process, measures applied should be reviewed. This stage should be executed periodically as defined in the design stage.

Establishing evaluation requirements and specifying the evaluation of the DataBench framework are briefly described in Table 2. The metrics are implemented in different dashboards which are revised periodically by the DataBench Administrators.

The monitoring and evaluation functionality of the DataBench framework will be represented in three different categories mapped to the user functionality and privacy criteria:

- Platform Metrics Dashboard
- User Metrics Dashboard
- Administrator Metrics Dashboard

The Platform metrics describe the key feature parameters of the DataBench framework that are used for static monitoring and evaluation. Examples for such metrics are total number of registered platform users, available registered benchmarks, number of use case scenarios, number of benchmark runs and others. These metrics will be available to all the different platform users to perform independent monitoring and evaluation of the platform environment.

User Profile metrics are generated for each specific user and describe his/her activities when using the platform. Example metrics are the number of benchmark searchers, number of downloaded benchmarks, number of submitted benchmark results and history log of all operations performed by the user in the last 30 days. These metrics will be used by both business and technical users to monitor their usage of the platform as well as to have a convenient history of the latest operations.

The Administrator metrics are in a way combination of the above two categories. The goal of this type of metrics is to enable the full monitoring of the DataBench framework from the static platform metrics to the user actions and operations performed in the different profiles. The administrator view will enable the performance of end-to-end platform analysis on the utilization of the platform. It will help to discover patterns and trends in the user searches and most executed operations.

In general all of the above type of metrics will be implemented in a separate dashboard and available according to the user credentials.

Table 2 describes briefly the mapping of factors to the stages of the evaluation process and the metrics to measure the quality factors.

Table 2. Example Metrics resulting from the DataBench Evaluation Process

Quality Factors	Stage 1 – Evaluation requirements	Stage 2 - Specify evaluation	Stage 4 – Execute and conclude evaluation
Effectiveness	What is the percentage of successful benchmark runs/executions?	Rate of successful tasks	Platform, User and Administrator Metrics Dashboard

Efficiency	What is the average time a user spends on actions in the DataBench website?	Average execution time of successful tasks	User and Administrator Metrics Dashboard
Satisfaction	How difficult was to work with the DataBench framework?	Single Ease Question	Administrator Metrics Dashboard
Performance efficiency	Are there sufficient hardware resources to host the DataBench service?	Resource utilization of hosting platform	Administrator Metrics Dashboard
Compatibility, Portability	On how many platform environments was the DataBench Toolbox executed?	Number of platform environments	User and Administrator Metrics Dashboard
Usability	How many benchmarks does a user execute on average?	Average of executed benchmarks per user	User and Administrator Metrics Dashboard
Usability	How many benchmark searches were executed? What are the most interesting benchmarks?	Number of benchmark searches	User and Administrator Metrics Dashboard
Usability	How many benchmarks results were submitted to the DataBench Toolbox?	Number of submitted benchmarks	Platform, User and Administrator Metrics Dashboard
Usability	How many use case scenarios are implemented?	Number of use case scenario.	Platform and Administrator Metrics Dashboard
Satisfaction, Usability, Functional suitability	What is the number of active users in the different roles?	Number of (active) users	Platform and Administrator Metrics Dashboard

Reliability	What is the time availability of the DataBench web service?	Uptime of DataBench Toolbox	Platform and Administrator Metrics Dashboard
Maintainability	How many executable benchmarks are offered in the DataBench Toolbox?	Number of implemented benchmarks	Platform, User and Administrator Metrics Dashboard
Maintainability	What is the average time necessary to register and integrate a new benchmark into the DataBench Toolbox?	Average time spent to add a new benchmark	Administrator Metrics Dashboard

4. DataBench Evaluation Implementation

This section revises the different DataBench Framework layers described in the previous deliverables with the goal to define how the evaluation and monitoring metrics will be integrated and implemented into the framework. The first part describes the changes in the general DataBench Methodology Workflow, followed by the new function modules in the DataBench Framework Architecture and the technologies used for the implementation of the three dashboards specified in the previous section.

4.1 DataBench Methodology Workflow

First, we revise and extend the DataBench Methodology Workflow presented in deliverable D1.1 [12], Subsection 4.1 from WP1 with new components necessary for the implementation of the evaluation process described in the previous Section 3.

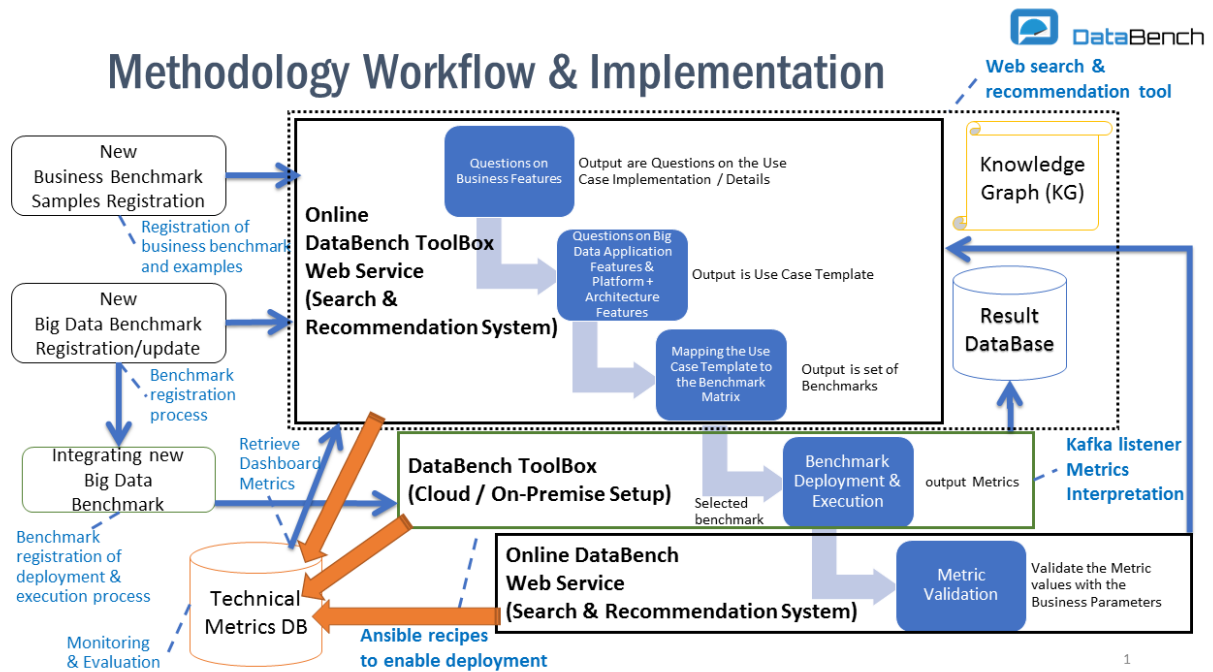


Figure 4. DataBench Methodological Framework

Figure 4 shows a schema of processes to illustrate how the different DataBench elements will support different set of users. A single user may have different roles, as identified in D3.1 [13], initially the following:

- **Benchmarking Providers:** Organization that owns a particular benchmark. They can be the actual developers of the benchmark or the organizations that maintain them. These users can register and update their benchmarks.
- **Technical Users:** Users that would like to search and potentially execute a technical benchmark. This includes the possibility of searching, downloading, executing and giving the results of the execution back to the Toolbox.

- **Business Users:** Users that would like to search and understand the business value of specific big data solutions. These users would not need to run technical benchmarks, but rather search for similar cases, business indicators, etc.
- **DataBench Admin:** People in charge of the administration of the Toolbox.

There are several processes depicted in Figure 4. On the left-hand side of the figure, the three boxes represent the registration process of two different kinds of benchmarks:

- The registration of data related to business-oriented big data benchmarks. The idea of the component located in the upper left corner of the figure (“New Business Benchmark Samples Registration”) is to capture domain and industry specific best practices and blueprints associated to concrete business KPIs.
- The registration of technical benchmarks. The two remaining components on the left represent the way the DataBench Toolbox will capture the necessary metadata and features about technical benchmarks to enable the search and recommendation processes (“New Big Data Benchmark Registration/Update” component), and to enable the automation of the deployment and the interpretation of the results of the execution of the benchmarks (“Integrating new Big Data Benchmark” component). Note that the registration of the automation provided by the second component is optional, in the sense that it requires the provision of deployment recipes and rules of interpretation of the results of the execution of the benchmarks which could prove a difficult task for some of the benchmarks analysed so far. However, the aim in DataBench is to automate as many as possible technical benchmarks, so the documentation of the process to integrate the automation will be also a key part for future extensibility to other benchmarks.

The components in the center of the Figure 4 show the full process from searching to executing and visualizing the results of benchmarks. The processes related to the DataBench Toolbox have been introduced in deliverable D3.1 [13] and D3.2 [14] will present the Alpha version of the Toolbox, while the validation of metrics is going to be introduced in deliverable D5.1 [15]. This process is divided into the following steps:

- **Search and Recommendation System:** The upper central box shows the steps to define the search criteria a user could pose to the system with the aim to select a benchmark that suits their needs. Based on those criteria (technical, business, application or platform features as explained in D1.1 [12], Section 3), the system will offer a set of potential benchmarks that could fulfil the user needs, as well as associated material (blueprints, best practices in sectors, etc.) that might facilitate the decision of the selection of the right benchmark.
- **The DataBench Toolbox setup:** The middle central box (in green in Figure 4) represents the process of deploying and enabling the execution either in cloud or in-premise of the selected benchmark. This could only happen if the registration of that benchmark provided the necessary recipes to allow the deployment. After the execution, the results of the benchmark will be sent back to the Toolbox for post-processing.
- **The validation of the metrics:** This process will allow in certain cases the matching of the technical metrics with business insights or KPIs. The results of the benchmarks will be then visualized and compared to others, giving the user a clear added-value

in comparison with the mere technical results that the execution of a technical benchmark may provide.

- **Monitoring and Evaluation:** This process gathers multiple metrics and internal component information with the goal to offer a monitoring and evaluation capabilities to the different users of the DataBench framework. All the gathered information is stored in a central Technical Metrics Database. The data is prepared, integrated, processed and visualized into a dashboard web service that can be accessed by the different users. The key functionality of this process is to enable both DataBench administrators, technical and business users to monitor how the DataBench framework evolves in time as well as perform an evaluation of the current framework state.

At the point of writing this document, partners are in the process of defining and prototyping the look and feel of the different processes listed in this section. The initial alpha version of the DataBench Toolbox is currently implemented and will be described in detail in deliverable D3.2 [14]. Similarly, the Metrics Validation process and the Knowledge Graph are under development and will be described in the upcoming WP5 deliverables.

4.2 DataBench Framework Architecture Modules

To realize the processes described in the DataBench workflow it was necessary to define and implement functional modules presented as part of the DataBench Framework Architecture in deliverable D3.1 [13] from WP3. The proposed modular framework is based on templates which are complemented with a web interface from where the user can decide

and choose the metrics needed. The web interface will also act as a dashboard where the results of the executions will be gathered and shown to the user, as seen in Figure 5.

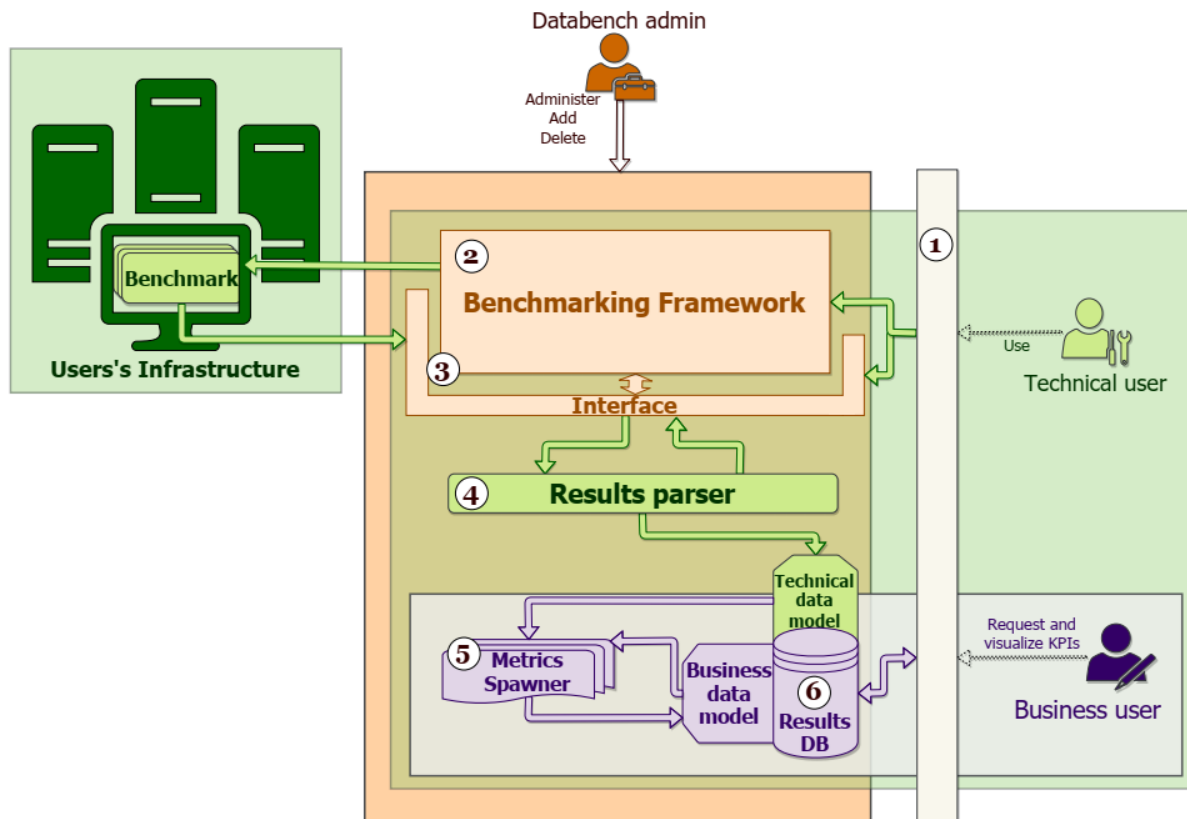


Figure 5. Functional Overview of the DataBench Framework Architecture [13]

The three Metrics Dashboards (Platform, User (Profile) and Administrator) will be also added to the web interface to support the evaluation and monitoring process introduced in Section 3. The proposed modular DataBench Framework Architecture is composed of the following interconnected modules:

1. Web interface;
2. Benchmark framework;
3. Results interface;
4. Results parser;
5. Metrics spawner;
6. Results DB;
7. **Metrics DB;**
8. **Metrics Dashboard.**

The first six are described in detail in deliverable D3.1 [13], whereas the Metrics DB and Metrics Dashboard are the new modules implementing the functionality necessary for the DataBench monitoring and evaluation process.

The Metrics DB module will be very similar to the Results DB module with the difference that it will store persistently the collected technical metrics. The goal is to reuse as much of the available functionality as possible, which means that the Metric spawner and the Results parser will be adapted to gather and prepare the metrics for the dashboards.

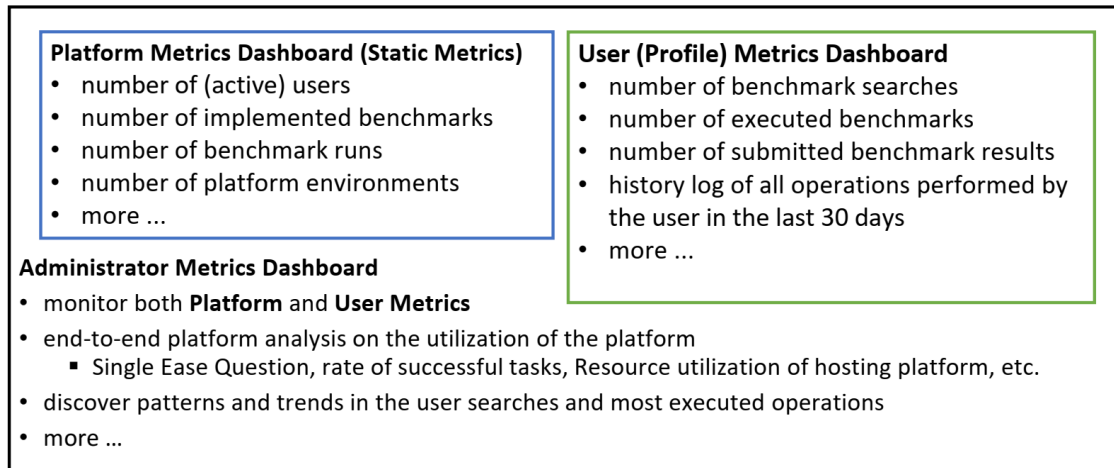


Figure 6. Overview of the Three Metrics Dashboards

The Metrics Dashboards will interface and visualize the different sets of metrics as depicted and summarized on Figure 6. Depending on the evaluation functionalities and user credentials, the collected metrics can be categorized in the following functional types:

- **Business metrics** are the one describing the business metadata, aggregated user data and all processing operations in the DataBench Framework.
- **Technical metrics** are the metrics gathered on the technical metadata (benchmarks), the aggregated benchmark results, user data and search results.
- **Static metrics** are the statistics extracted from the metadata that is changed unfrequently and mostly by the DataBench administrators.
- **Dynamic metrics** are the active and operational statistics, which are aggregated in the process of benchmark result submission, user searches and analysis of use cases and results.
- **Anonymized metrics** are statistics which can be shown directly to the DataBench users and are not including any sensitive private or security data.
- **Raw metrics** are statistics which are freshly gathered by the DataBench tool and are not anonymized or prepared for external users, and include sensitive private data and similar.

4.3 DataBench Framework Technology Implementation

After defining the functional modules in the DataBench Framework, it is essential to explain how the technical implementation is realized. The DataBench Alpha version of the Toolbox is implemented on a set of pre-existing frameworks and tools that allow not starting from scratch to develop the desired functionality and that can be integrated with the Ansible [16] approach for deployment. The implementation details are explained in deliverable D3.2 [14] and depicted on Figure 7, which shows the main frameworks and their interaction that serve as baseline for the different elements of the architecture.

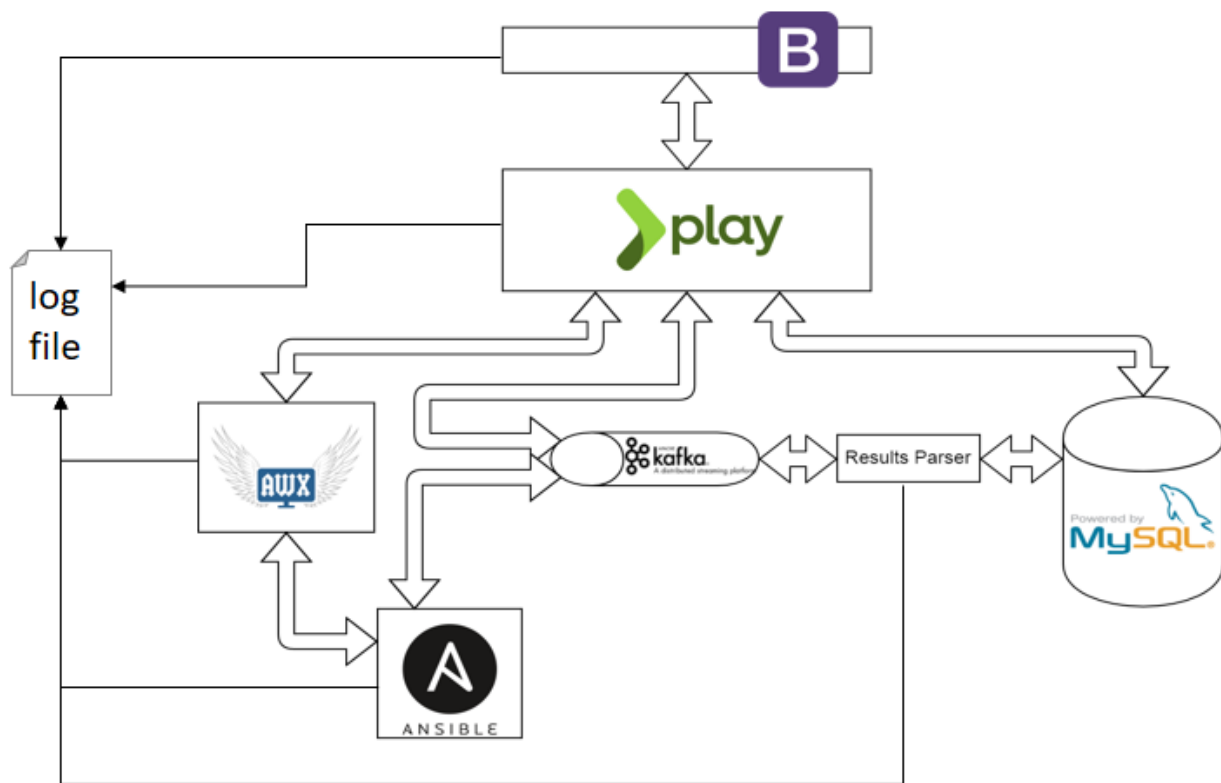


Figure 7. Overview of the technical implementation of the Alpha version [14]

The main core of the Toolbox is hidden behind the scenes. The backend is what provides the full range of options available to run the different integrated benchmarks. It is comprised of several modules described below:

- Bootstrap [17]: The GUI of the Alpha version has been developed using the Bootstrap framework. Bootstrap is a toolkit that allows the creation of powerful web applications based on HTML, CSS and JavaScript. It provides many open templates that facilitates the development of webs as well as many out-of-the-box graphical elements to ease the process of web development.
- Play!-Framework [18]: It is the backend framework used to implement the web functionality. Using Java or Scala as programming language, it provides a MVC (Model-view-controller) development pattern. It also supports easy integration with Bootstrap and different databases through JDBC making it suitable for the project.
- AWX project [19]: AWX is the upstream open source project of Ansible Tower, which allows the control of the automation of deployment of software and tools. AWX provides a web dashboard, a rest API and a task engine on top of Ansible [16].
- Kafka [20]: It is a well-known distributed streaming platform based on the public subscription paradigm. It is used to act as an interface between Ansible and the Results database. Kafka is used in the Alpha version as the middleware to get the results from the execution of the benchmarks (publisher) and the Results parser (subscriber).
- MySQL [21]: Widely used Open source relational database management system. The Alpha version is using MySQL as backend because the data generated and stored in the platform is mostly of relational structure, thus making it easier to store it in a relational

database. Future versions of the Toolbox will interact with the Knowledge Graph developed in the scope of WP5 and described later in Section 5.

- Log file: With the introduced metrics, a new central data store for all the different sources is necessary. As a starting point, all the actions performed on the DataBench framework logs the executions in a file. A periodically process extracts the essential information of the log file and stores the data in the Metrics DB. The metrics of the dashboards will mainly use the retrieved data which is stored in the database. Also, it is still possible to interact with the different technologies directly by using their APIs.

As already explained in Section 3, we define three categories of Metrics Dashboards: Platform, User (Profile) and Administrator. These dashboards will be implemented using the above technologies and the data that is already available in the framework modules. Mostly data of Ansible, Bootstrap, and the MySQL database are used. Below is a list of the relevant metrics implemented in each of the dashboards avoiding the repetition of metrics in the administrator dashboard:

List of Platform Metrics:

- Rate of successful tasks
- Number of (active) users
- Uptime of DataBench Framework
- Number of Implemented Benchmarks

List of User (Profile) Metrics:

- Average Execution Time of Successful Tasks
- Number of Platform Environments
- Number of Average Executed Benchmarks per User

List of Administrator Metrics:

- Single Ease Question
- Resource Utilization of Hosting Platform
- Average Time Spent to add a New Benchmark

A first mockup of a dashboard is shown in Figure 8 depicting as an example the Platform Metrics Dashboard.

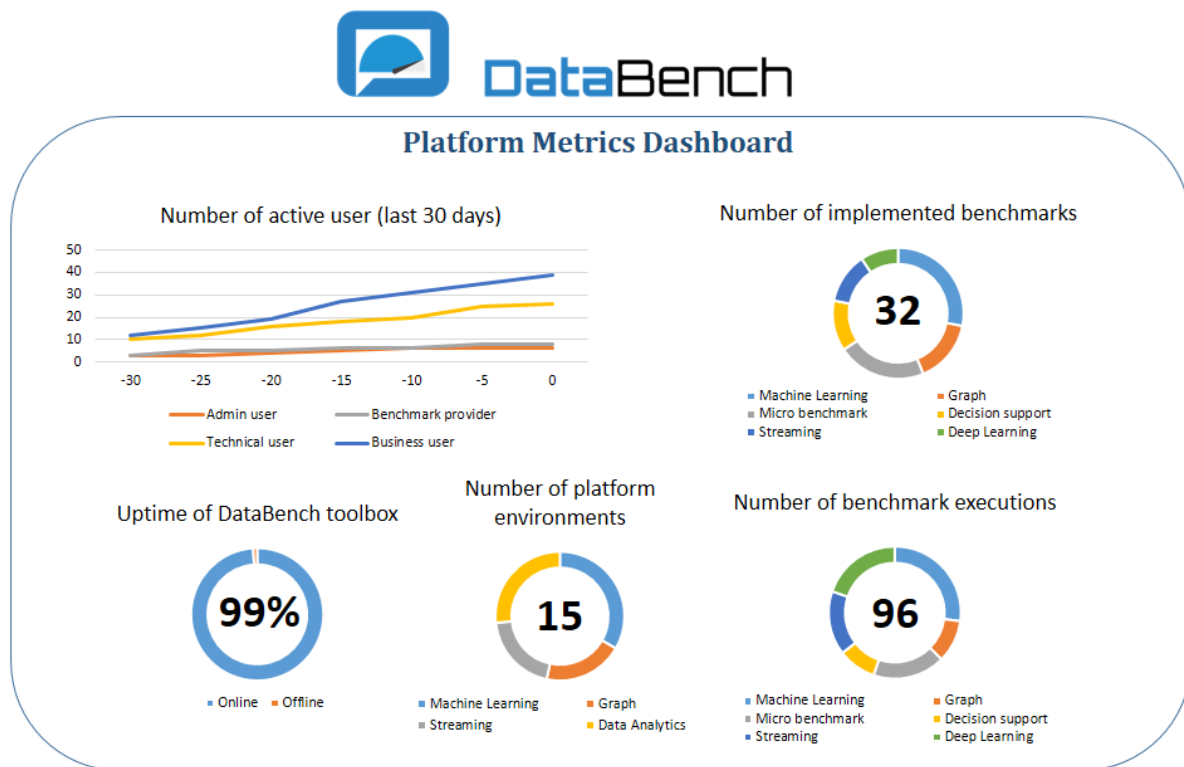


Figure 8. Example for Platform Metrics Dashboard

5. DataBench Knowledge Graph Monitoring

DataBench Deliverable D1.1 [12] states that the information collected at various stages of the project will be organized in a form to be easily accessible, structured and interoperable with other semantic knowledge resources.

For that purpose we plan to use a recently popular data structure called ‘Knowledge Graph’ (KG) [22] allowing flexible data schemas and be scalable for operations like search, aggregation, and in particular interlinked with other relevant global semantic vocabularies and resources like WikiData [23] and LinkedOpenData [24].

The DataBench Knowledge Graph would be targeted at different user groups, including **Business Users** who would like to search and understand the business value of big data approaches and appliance of big data in the production line. **Technical users** and **Benchmark providers** can use DataBench Knowledge Graph as solution for tracking the novelties associated with Big Data technologies and Benchmarking.

The DataBench Knowledge Graph should include information available from the:

- Questionnaires (in particular, user survey conducted in WP2);
- Interviews with companies adopting Big Data technologies;
- Descriptions and analysis of Benchmarking tools provided by DataBench partners;
- Technical results from benchmarking experiments (where a set of KPIs are measured and can be uploaded into the Knowledge Graph);
- Literature review associated with big data technologies, data science, benchmarking and other relevant topics. In particular, the regular automatic literature analysis can provide a possibility to capture and describe in the Knowledge Graph the novel data science algorithms and tools, along with relevant datasets mentioned in the literature.
- Other reports and available information about big data benchmarking, including but not limited to: reports from analysts, big data solutions applied in specific domains or usage scenarios, use case studies, etc.

In order to perform literature analysis and review JSI uses the Microsoft Academic Graph and available Open Education Resources in the area of big data, data science, benchmarking and relevant technologies to create an automatically refreshed database of newest methodologies, tools and datasets.

The Microsoft Academic Graph [25] is a heterogeneous graph containing scientific publication records, citation relationships between those publications, as well as authors, institutions, journals, conferences, and fields of study. This graph is used to power experiences in Bing, Cortana, Word, and in Microsoft Academic. The graph is currently being updated on a weekly basis.

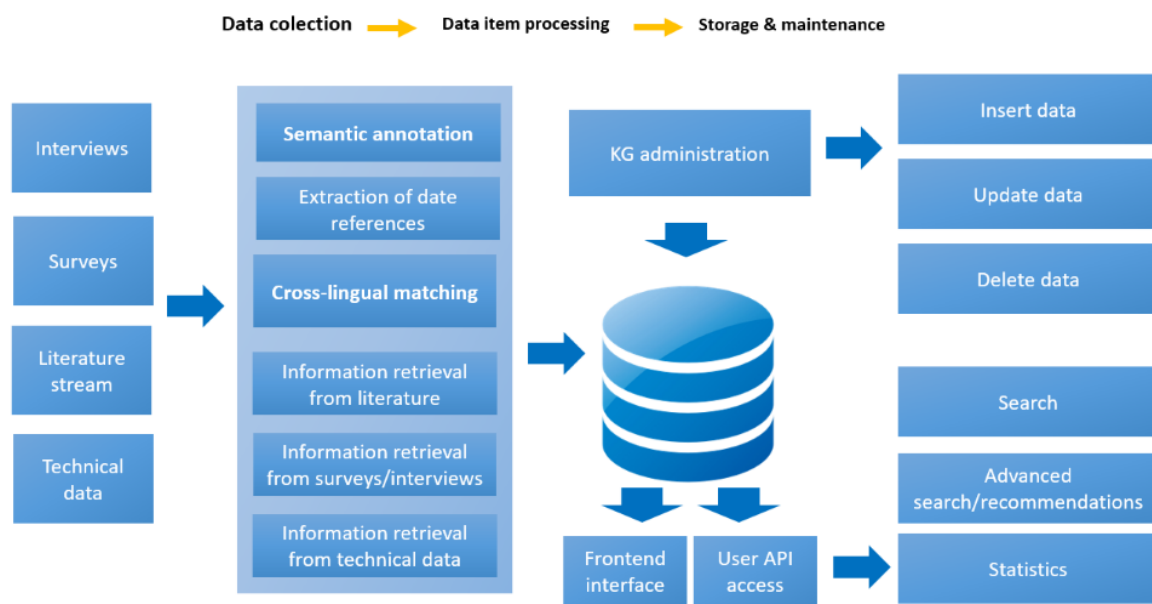


Figure 3. DataBench Knowledge Graph Pipeline

Figure 3 shows the initial pipeline for DataBench Knowledge Graph component. The Knowledge Graph should interact with the Toolbox and the Web Service via specific API, that would provide a number of search and analytics options for the users, including:

- the possibility to search on topic level;
- the possibility to combine a number of search options in the advanced search;
- the statistics/aggregation functionalities based on features extracted from the literature stream, surveys and interviews and technical data.

The admin functionalities would include the options to insert, update and delete data (such as technical data obtained from Toolbox or literature stream).

6. Conclusion

This document provides an overview of the DataBench framework evaluation and monitoring methodology as part of the technical analysis in Task 5.2 of Work Package 5. It presents an initial approach for evaluation and assessment based on the current ISO/IEC standards for system and software quality. The defined quality factors are mapped to technical metrics that will be implemented as part of the DataBench framework and cover various technical characteristics. To achieve this goal with the least possible effort, we revise the DataBench Workflow Methodology and Architecture Modules to extend the existing modules and reuse the technology frameworks already utilized in the DataBench Toolbox. The resulting web interfaces will depict three different (Platform, User(Profile) and Administrator) Metrics Dashboards that will report relevant role-based metrics, with which the user can assess and analyse his actions and obtained results. The dashboards will guarantee the DataBench framework usability, relevance, scale and operations transparency. The data foundation (e.g. log info of user interaction, Ansible statistics, etc.) and the dashboards will be implemented in the following releases of the DataBench framework and Toolbox. New metrics will be implemented and added as the framework implementation progresses.

Overall, by adapting the ISO/IEC system and software quality standard characteristics into our evaluation approach, we ensure that the DataBench framework comply to the latest software usability and evaluation standards.

References

- [1] A. Abran, A. Khelifi, W. Suryn, and A. Seffah, "Usability Meanings and Interpretations in ISO Standards," *Softw. Qual. J.*, vol. 11, no. 4, pp. 325–338, Nov. 2003.
- [2] J. Estdale and E. Georgiadou, "Applying the ISO/IEC 25010 Quality Models to Software Product," in *Systems, Software and Services Process Improvement - 25th European Conference, EuroSPI 2018, Bilbao, Spain, September 5-7, 2018, Proceedings*, 2018, vol. 896, pp. 492–503.
- [3] Y. Zhao, J. Gong, Y. Hu, Z. Liu, and L. Cai, "Analysis of quality evaluation based on ISO/IEC SQuaRE series standards and its considerations," in *16th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2017, Wuhan, China, May 24-26, 2017*, 2017, pp. 245–250.
- [4] "ISO/IEC 25000:2014(en), Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE." [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso-iec:25000:en>. [Accessed: 11-Jun-2019].
- [5] K. Esaki, "System quality requirement and evaluation, importance of application of the ISO/IEC25000 series," *Glob. Perspect. Eng. Manag.*, vol. 2, no. 2, pp. 52–59, 2013.
- [6] A. Hussain and E. O. C. Mkpojiogu, "AN APPLICATION OF THE ISO/IEC 25010 STANDARD IN THE QUALITY-IN-USE ASSESSMENT OF AN ONLINE HEALTH AWARENESS SYSTEM," *J. Teknol.*, vol. 77, no. 5, Nov. 2015.
- [7] S. Ouhbi, A. Idri, J. L. Fernández-Alemán, A. Toval, and H. Benjelloun, "Applying ISO/IEC 25010 on Mobile Personal Health Records," in *Proceedings of the International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5*, Portugal, 2015, pp. 405–412.
- [8] O. Canalejo and M. Isabel, "A Quality Model for Big Data Database Management Systems," 2018.
- [9] H. Nakai, N. Tsuda, K. Honda, H. Washizaki, and Y. Fukazawa, "A SQuaRE-based software quality evaluation framework and its case study," in *2016 IEEE Region 10 Conference (TENCON)*, 2016, pp. 3704–3707.
- [10] N. Tsuda *et al.*, "WSQF: Comprehensive Software Quality Evaluation Framework and Benchmark Based on SQuaRE," in *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice*, Piscataway, NJ, USA, 2019, pp. 312–321.
- [11] "ISO/IEC 13236:1998(en), Information technology — Quality of service: Framework." [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso-iec:13236:ed-1:v1:en>. [Accessed: 27-Jun-2019].
- [12] DataBench Project, "Deliverable D1.1 Industry Requirements with benchmark metrics and KPIs." .
- [13] DataBench Project, "Deliverable D3.1 DataBench Architecture." .
- [14] DataBench Project, "Deliverable D3.2 DataBench Toolbox - Alpha including Support for Reusing of Existing Benchmarks." .

- [15] DataBench Project, "Deliverable D5.1 Initial Evaluation of DataBench Metrics." .
- [16] A. Hat Red, "Ansible is Simple IT Automation." [Online]. Available: <https://www.ansible.com>. [Accessed: 17-Jun-2019].
- [17] M. O. contributors Jacob Thornton, and Bootstrap, "Bootstrap." [Online]. Available: <https://getbootstrap.com/>. [Accessed: 17-Jun-2019].
- [18] "Play Framework - Build Modern & Scalable Web Apps with Java and Scala." [Online]. Available: <https://www.playframework.com/>. [Accessed: 17-Jun-2019].
- [19] A. Hat Red, "Open Sourcing Ansible Tower." [Online]. Available: <https://www.ansible.com/products/awx-project>. [Accessed: 17-Jun-2019].
- [20] "Apache Kafka," *Apache Kafka*. [Online]. Available: <https://kafka.apache.org/>. [Accessed: 17-Jun-2019].
- [21] "MySQL." [Online]. Available: <https://www.mysql.com/>. [Accessed: 17-Jun-2019].
- [22] "Knowledge Graph," *Wikipedia*. 10-Jun-2019.
- [23] "Wikidata," *Wikipedia*. 13-May-2019.
- [24] "Linked Data | Linked Data - Connect Distributed Data across the Web." [Online]. Available: <http://linkeddata.org/>. [Accessed: 17-Jun-2019].
- [25] "Microsoft Academic Graph," *Microsoft Research*. .