VISDOM
Visual software diagnostics

D3.4.2 Second research demo that shows aspects of one use case and first version of the configurable dashboard

Programme	ITEA3		
Challenge	Smart Engineering		
Project number	17038		
Project name	Visual diagnosis for DevOps software development		
Project duration	1st October 2018 – 30st June 2022		
Project website			
Project WP	WP3 - Visualizations		
Project Task			
Deliverable type		Doc	Textual deliverable
	X	SW	Software deliverable
Version	V1.0		
Delivered	04/02/2021		
Access	x	Public	
		Abstracts are public	
		Confidential	

Document Contributors		
Company	**Author**	**Role**
EXPERIS	Ester Sancho	writer/editor
EXPERIS	Gema Maestro	writer
UPC	Lidia López	writer
TAU	Kari Systä	writer
TAU	Outi Sievi-Korte	writer
TAU	Vivian Lunnikivi	writer
U.Oulu	Henri Bomström	writer
TIOBE	Marvin Wener	reviewer

Document History			
Date	**Version**	**Editors**	**Status**
16.11.2020	ToC	EXPERIS	Table of Content
19.11.2020	V01	EXPERIS	Draft
13.12.2020	V02	TAU	Teaching demo
21.12.2020	V03	UPC	Quality demo
15.01.2020	V04	EXPERIS	Final Version
28.01.2021	V05	TIOBE	Reviewed Version
01.02.2021	V1.0	EXPERIS	Submitted Version
04.02.2021	V1.1	EXPERIS	Fixes
04.02.2021	V1.2	TAU	Added Refences

Table of Contents

Table of Contents 3
1. Introduction 4
2. Introduction to Demonstrations 5
   2.1. Quality use case 5
   2.2. Teaching use case 5
3. Demo for Uses Cas 1 - product quality 6
   3.1. Tailoring Quality Indicators 6
   3.2. Quality Visualisations 7
   3.3. Dashboard Configuration 11
4. Demo for Use Case 2 - Teaching Software Development 13
   4.1. The configurable Dashboard 13
   4.2. Utilizing Composer in the Teaching Use Case 13
   4.3. Visualizations in the Teaching Use Case 13
   4.4. Stakeholder Feedback on the Demo 15
5. Access to the demos 17
6. Next Steps & Conclusions 17

Table of Figures

Figure 1: Defining a quality factor in the quality use case. 7
Figure 2: Quality dashboard main view. 7
Figure 3: Detailed Strategic Indicators view (radar). 8
Figure 4: Detailed Strategic Indicators view (polar). 8
Figure 5: Detailed Strategic Indicators view (stacked). 9
Figure 6: Quality indicators evolution. 10
Figure 7: Strategic Indicators Heatmap 10
Figure 8: Quality prediction view 11
Figure 9: Quality profiles configuration form. 12
Figure 10: Quality categories for strategic indicators (quality use case). 12
Figure 11: The default dashboard for teachers, showing student progress over the course. 14
Figure 12: The current default dashboard for teaching assistants showing current status for each student. 15
Figure 13: Raw Data Screen for Product Quality Demo 17

1. Introduction

The main objective of WP3 is to define and implement graphical representation (visualizations) of software projects, using the data collected and analysed from those software projects and artefacts. Seeking to deliver new visualizations and configurable dashboards that can contain several visualizations oriented to different types of profiles or users, with different needs. In order to achieve these objectives, a thorough analysis of the state of the art has been carried out mainly in the areas of visualization, data extraction and processing in software projects (D1.1.1), and finally in the appropriate techniques and methods for implementing visualization in the context of DevOps (Development & Operations) (D.3.1.1).

During the definition and implementation of visualizations, the VISDOM-project produces a set of demos that are used for both internal and external communication. These demos or proof of concepts presented in this deliverable will be exposed to the stakeholders on a regular basis. All feedback obtained will be translated into more elaborate demonstrations that will be tailored to the stakeholders needs and concerns.

VISDOM includes different company cases. Previous studies have been carried out in parallel with the corresponding agents involved. Several types of users were identified in these studies. Their needs are complementary in the framework of VISDOM, therefore it was considered to present two demos in this deliverable, each of them tailored to a specific user.

The main content of the deliverable is composed from two concrete demonstrations.

1. Quality case: a configurable dashboard for monitoring software quality of DevOps projects.
2. Teaching case: a configurable dashboard for teachers of a programming course.

This document gives the background of those demonstrators. Links to the actual demonstrators are given at the end of the document.

As mentioned above, within these use cases, we can differentiate between two types of users depending on the use of the tool and the user's needs. On the one hand, we have users who need support for decision-making, which means they do not need to explore the data but simply to be able to instantly understand the information, without going into detail. These same users (or their organisation) may not have the technological knowledge to develop or implement specific charts. On the other hand, we have users with specific visualisation needs and with the necessary skills to generate and code these types of visualisations. These users intend to analyse the data in detail and in a specific way.

These demos have been conceived as complementary with possibilities of future integration from a conceptual point of view.

2. Introduction to Demonstrations

2.1. Quality use case

The quality demo builds around a DevOps team that develops a product named PHE developed in Experis (that consists in health & wellbeing services for their employees) composed of two projects. These two projects consist of: (a) an application (APP) to be deployed in the final users devices (named phe); and, (b) a set of services to be deployed in a server that are consumed by the APP (named phe_server). Following DevOps strategy, the team is composed of two connected teams, development team that is responsible for the development of both products and operation and a team that is focused on the services deployment. The source code and the backlog are managed using GitLab. GitLab data is complemented by static (source) code analysis provided by SonarQube and the continuous integration tool Jenkings.

In the context of quality management of software product development, the main stakeholders are the project manager, product owners, a system analyst, developers and server technicians.

Based on a workshop conducted in July 2019, we identified their different needs of data visualization. In this use case, all the stakeholders need the same kind of visualizations, they need visualizations that allow them to see if a specific quality-related property is going well, like a quality indicator (good/moderate/poor). In this workshop, we identified a set of indicators to be included in the demonstration. The project manager (responsible for both products) was seeing the progress and efficiency of the development and the quality of the final product. The quality of the product depends on the product, for the development team they are interested in the quality of the product in terms of bugs and for the operations team in terms of availability and performance.

2.2. Teaching use case

The teaching demo builds around a programming course, taught in Tampere University. The course work consists of weekly exercises, project works and an exam, and all course activities grant students points towards the final course grade. Since there are usually from 100 to 300 students on the course at a time, all the course material and exercise assignments as well as their submission boxes for automatic grading are published in a learning management system called Plussa. Students solve weekly exercises locally and push their solutions into their own code repositories, hosted in the University's GitLab instance. Therefore, GitLab works as the main source of automatically collected data for the teaching demo, and Plussa serves as another source.

In the teaching use case, the identified main stakeholders are teachers, teaching assistants and students. The teacher on a course is responsible for the course arrangements and might hire teaching assistants, who help with grading projects or provide support for students in completing weekly assignments. Students work toward completing the course by learning the topics and practicing the required skills by solving the exercises. As these stakeholders have differing tasks and responsibilities on the same course, they have individual information needs and thus, are presented with different views into the data.

Based on a stakeholder interview, the main interest for teachers was identified to be seeing how students' progress over a course. Seeing the progress helps teachers to evaluate if there are, for example, some difficult topics or bottleneck exercises that require extra attention. However, teaching assistants are more interested in seeing the current status of students during each course week, to be able to identify students that might need help. Students, on the other hand, are likely interested to see how they are progressing in relation to the course goals and how much work they are required to do to achieve their personal goals. Visualizing progress in relation to course requirements helps students plan their work and possibly even motivate them.

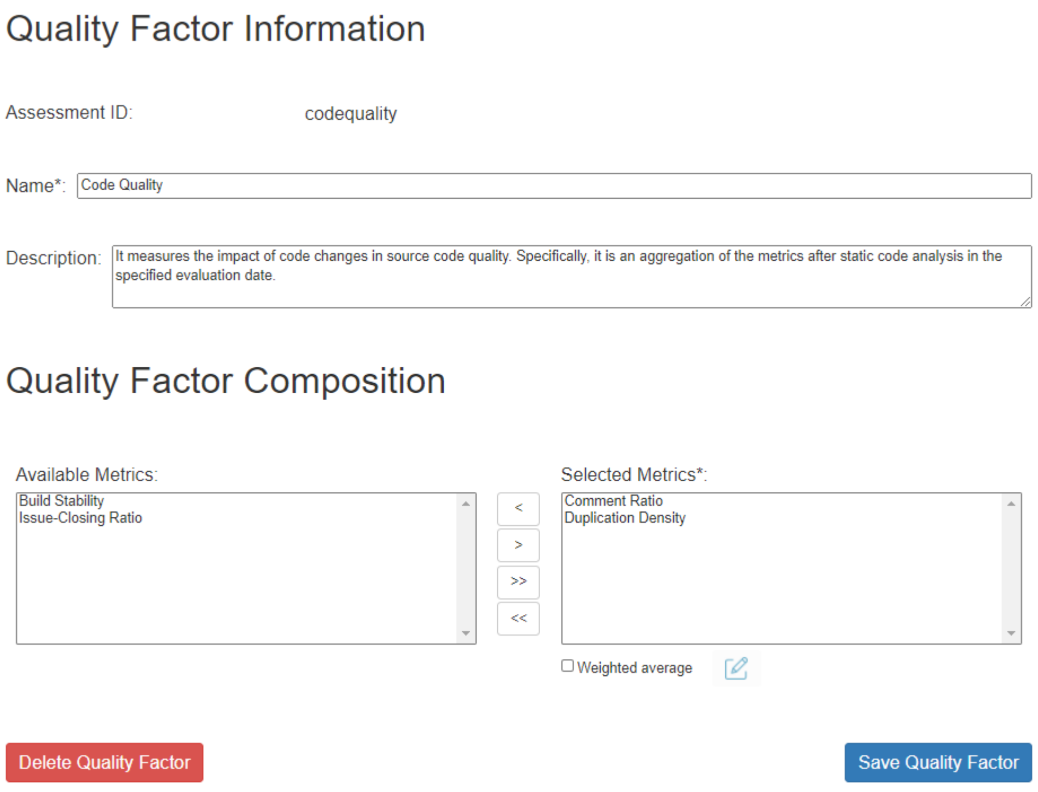
3. Demo for Uses Cas 1 - product quality

The quality demo consists of one software component with three main functionalities: tailoring quality indicators, visualizing quality, and configuring the dashboard. The first two functionalities are devoted to regular users (project manager, product owner, system analyst, developer and server technician), and the third is for system administrators.

3.1. Tailoring Quality Indicators

In the dashboard, the quality indicators are defined as aggregations-based metrics computed directly from the data provided from the data source tools (see Deliverable 2.3.2 for details). These metrics are aggregated in quality factors, addressing concrete quality attributes. These quality factors can be aggregated into the final quality indicators, that are named strategic indicators.

The dashboard provides forms for defining dedicated quality attributes and strategic indicators of each product to be assessed (phe and phe_server). Figure 1 shows the form for creating the quality factor *Code Quality*. It is computed based on the metrics related to static code analysis *Comment Ratio* and *Duplication Density*. The dashboard shows the list of available metrics in the left list, and the user can select and move to the right list the ones he or she wants to use for the quality factor computation. This aggregation can be calculated as an average or a weighted average.


Quality Factor Information

Assessment ID:	codequality
Name*:	Code Quality
Description:	It measures the impact of code changes in source code quality. Specifically, it is an aggregation of the metrics after static code analysis in the specified evaluation date.

Quality Factor Composition

Available Metrics:	Selected Metrics*:
Build Stability	Comment Ratio
Issue-Closing Ratio	Duplication Density

Figure 1: Defining a quality factor in the quality use case.

The dashboard includes the same kind of form to define strategic indicators, for strategic indicators, the user needs to select from the list of available quality factors.

3.2. Quality Visualisations

When the strategic indicators and quality factors are defined, the dashboard provides several ways to visualise these indicators. Figure 2 shows the main view when the dashboard is open.

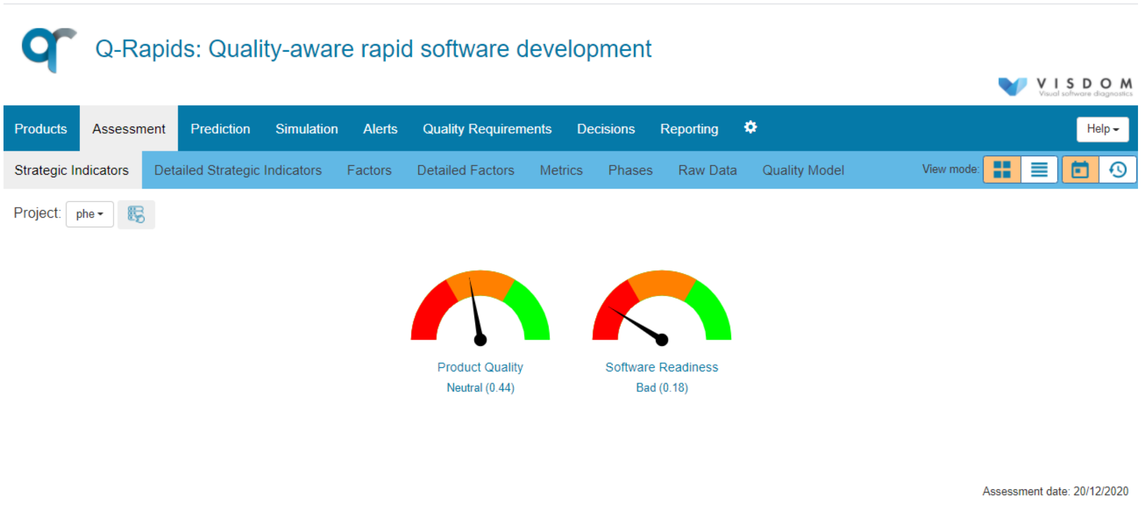


Figure 2: Quality dashboard main view.

On the menu, the user can select what to do (first menu level) and what to see (strategic indicators, quality factors, and metrics). Figure 2 is showing the strategic indicators assessment for today providing this good/moderate/poor views through gauge charts. The categories, and their colours, are configurable. In the Experis use case, we configured quality as 3-levels: poor (red), moderate (orange), and good (green), the qualification for each level is configurable. This view shows a moderate quality for product quality and poor quality for software readiness, the software is not ready to release yet. The dashboard also allows the user to visualise the quality factor used in the assessment of the quality factors, for visualising this detailed view, the dashboard provides three different visualisations: using radar charts (see Figure 3), using polar charts (Figure 4), and stacked bar charts (Figure 5).

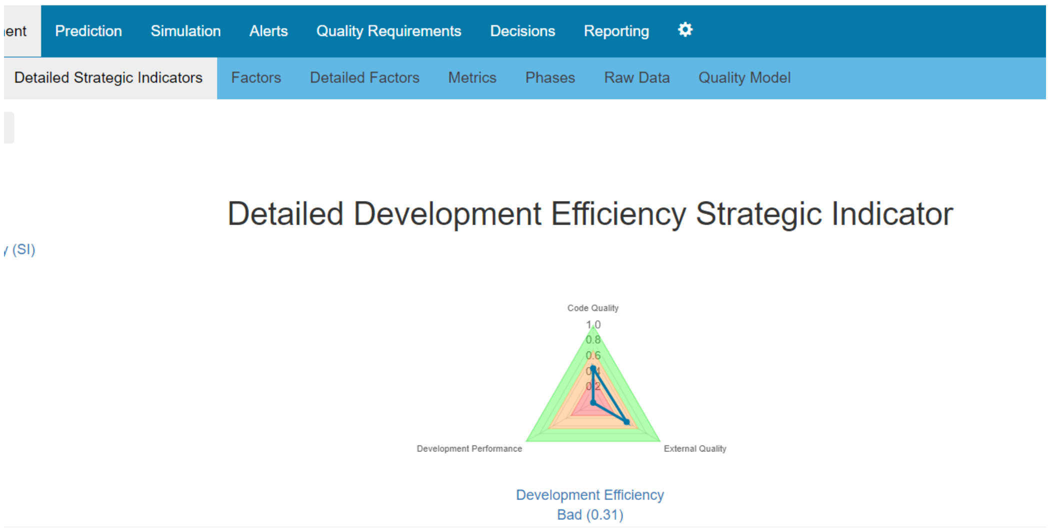


Figure 3: Detailed Strategic Indicators view (radar).

Radar charts visualizes the quality assessment for the quality factor. For example, for *Release Readiness* indicator, *Build Status* factor value is 0.35 and *Development status* is 0. Therefore, they need to devote more resources to improve their development status.

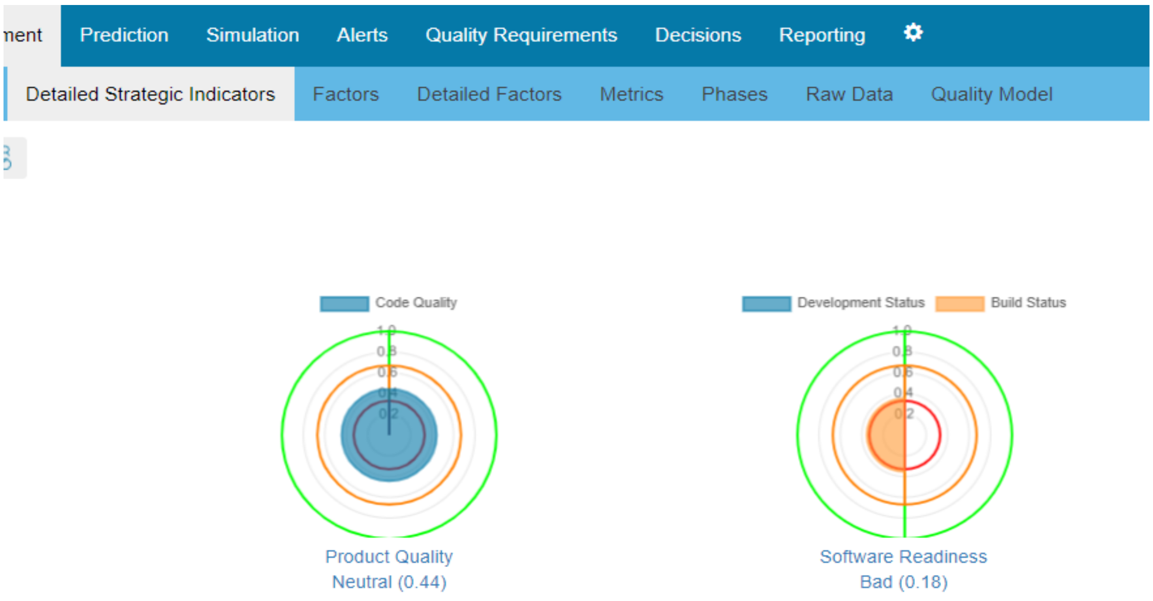


Figure 4: Detailed Strategic Indicators view (polar).

Polar and stacked bar charts show how the quality factor value is contributing to the strategic indicator value. For example, *Code Quality* is the only factor for measuring *Product Quality*, therefore, the *Product Quality* indicator value is the same as *Code Quality* (0.44). In the example of *Software readiness*, *Build status* corresponds to 0.18 to *Software Readiness* because *Software Readiness* has two factors (50%+50%), therefore 0.35*0.5 is equals to 0.175 (rounded to 0.18 in the chart).

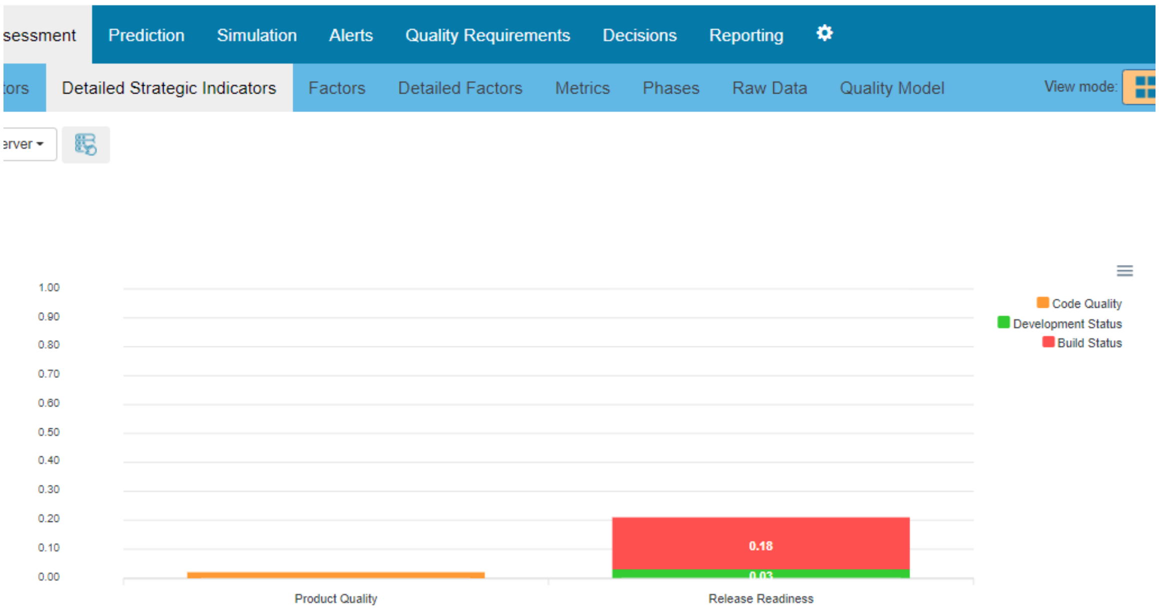


Figure 5: Detailed Strategic Indicators view (stacked).

These visualizations also include information about the quality categories in the charts (poor-red, moderate-orange, and good-green), being represented by areas (e.g. gauge, radar) or lines (polar). The dashboard provides the same kind of visualizations for quality factors and metrics.

Dashboard also provides the possibility of visualising the evolution of these indicators (strategic indicators, quality factors, and metrics) using line charts. Figure 6 shows the kind of charts used to visualise the evolution of the strategic indicator Release Readiness (left) and the evolution of the quality factors *Build Status* and *Development status* used to compute it (right). In these charts, the user can see a stable quality assessment, no changes in the last two weeks.

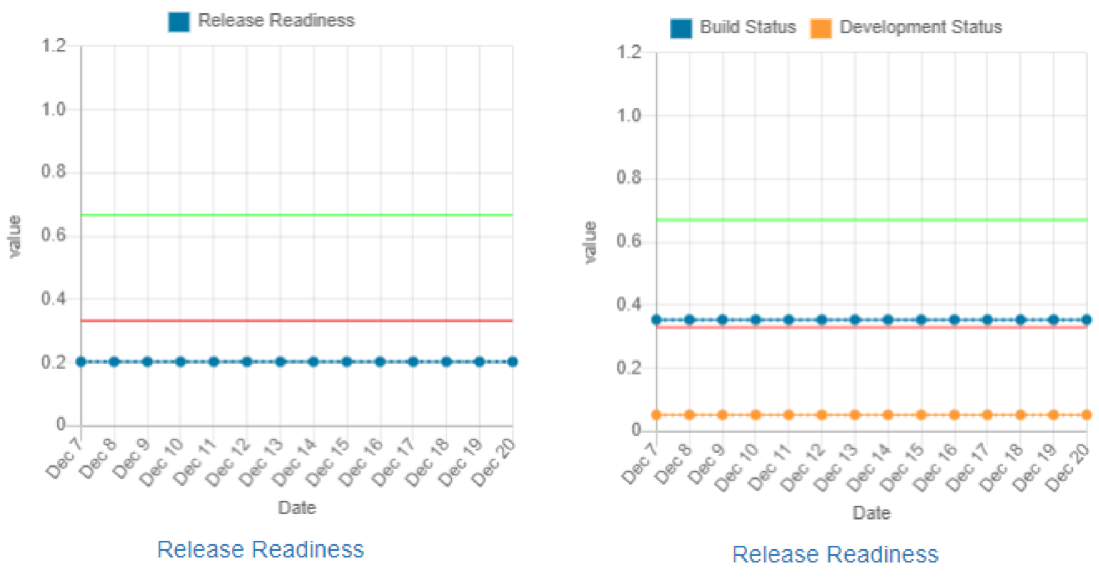


Figure 6: Quality indicators evolution.

Heatmaps are generally used for hotspot identification like this file has many bug fixes. In this case, we considered using this kind of visualization to display the evolution of the strategic indicators summarised by periods (Figure 5).

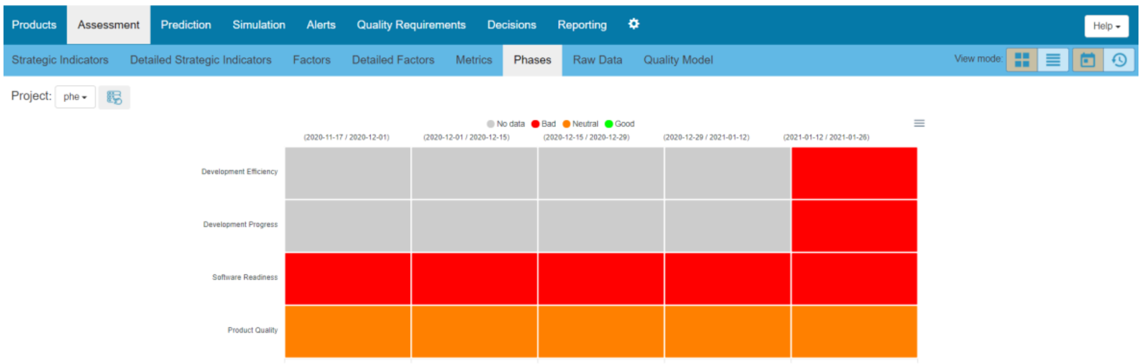
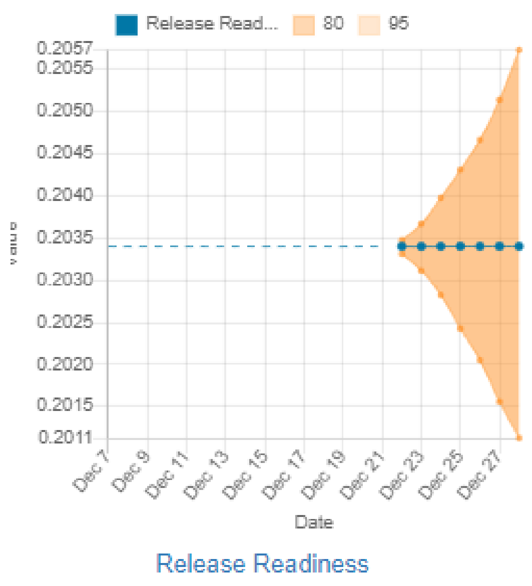


Figure 7: Strategic Indicators Heatmap

Quality dashboard also includes a view showing the quality prediction. Figure 8 shows the prediction for the strategic indicator Release Readiness for the following two weeks. These charts include some historical data (dotted line), the predicted values (solid line chart), and the confidence intervals corresponding to 80 and 95 percent of confidence. On the left, we are visualising the predicted values for the strategic indicators, on the right the predicted values for the quality factors for the strategic indicator.



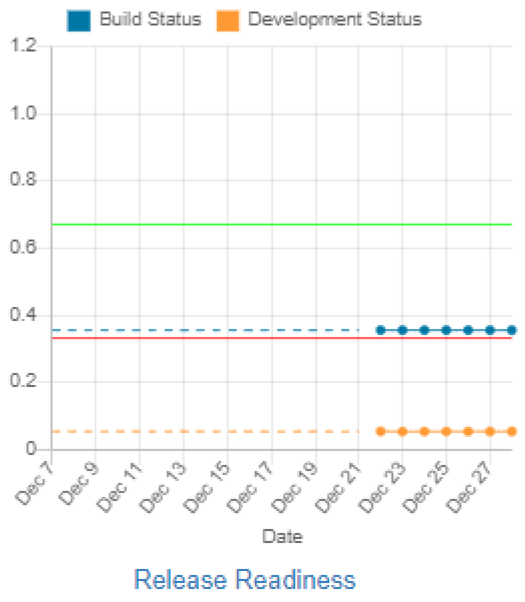


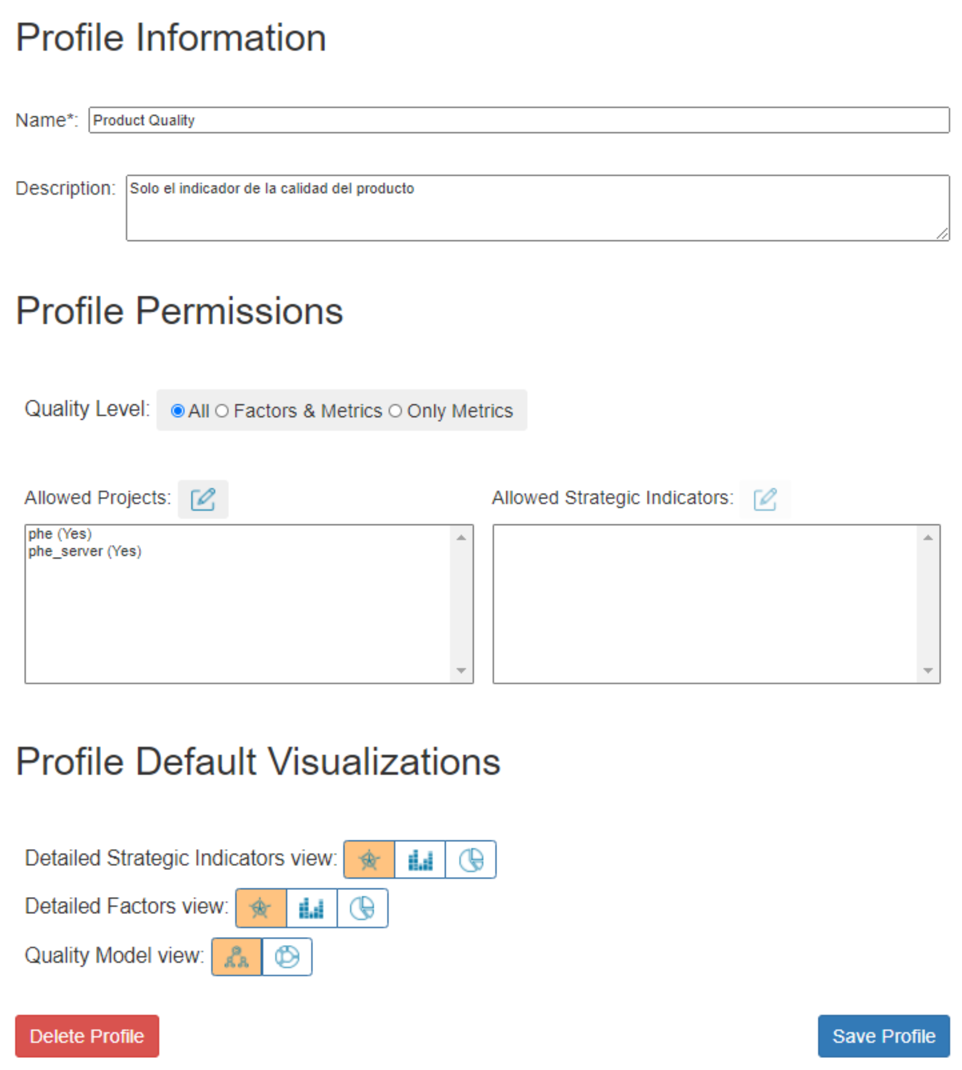
Figure 8: Quality prediction view

3.3. Dashboard Configuration

As we explained in section 2.1, all the stakeholders need the same kind of visualizations. Therefore, in this use case, the configuration needs are related to what data should be visualised by the dashboard.

The dashboard provides a profiles manager, the system administrator can create as many profiles as needed. Figure 9 shows the profile configuration form, each profile determines:

- *Quality level.* The quality indicators managed by the quality dashboard (from higher to lower level) are strategic indicators, quality factors, and metrics. The profile can restrict the access to some levels, e.g, only metrics.
- *Allowed projects*. The dashboard allows to visualise the quality indicators for several projects. In the Experis case, they decided to monitor phe and phe_server projects, the profile can provide access to phe, to phe_server, or both.
- For each project, the *allowed strategic indicators*. The dashboard allows to define several strategic indicators for each project. In the Experis case, they defined the same two indicators for both projects: *Project Quality* and *Release Readiness*, depending on the profile, the user will be able to monitor one of each or both.
- *Default visualizations*. For some of the quality views, the dashboard provides several views, in this deliverable, we show three different views for the detailed strategic indicators (i.e., using radar, polar, or stacked bar charts). The profile defines which kind of charts should be shown.



Profile Information

Name*: Product Quality

Description: Solo el indicador de la calidad del producto

Profile Permissions

Quality Level: [✓ All] Factors & Metrics Only Metrics

Allowed Projects:

phe (Yes)
phe_server (Yes)

Allowed Strategic Indicators:

Profile Default Visualizations

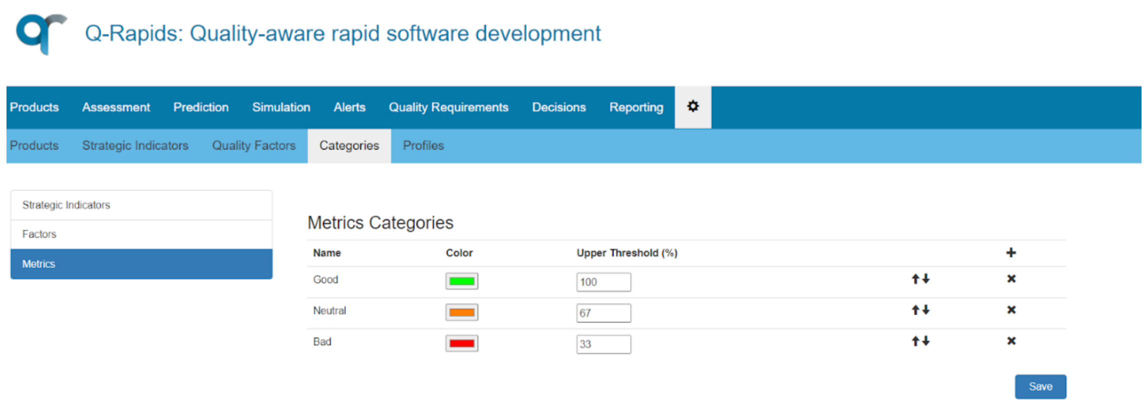
Detailed Strategic Indicators view:

Detailed Factors view:

Quality Model view:

Figure 9: Quality profiles configuration form.

Figure 10 shows the configuration form for defining the quality categories for strategic indicators. The number of categories, colours, and thresholds can be different for strategic indicators, quality factors, and metrics.



Metrics Categories

Name	Color	Upper Threshold (%)		+
Good	🟩	100	↑↓	×
Neutral	🟧	67	↑↓	×
Bad	🟥	33	↑↓	×

Figure 10: Quality categories for strategic indicators (quality use case).

4. Demo for Use Case 2 - Teaching Software Development

The teaching demo consists of two main software components: the configurable dashboard and independent visualization applications that are imported to the dashboard service.

4.1. The configurable Dashboard

An implementation of a configurable dashboard is being developed as a part of a Composer service under construction by the team in University of Oulu. The deployed version of the Composer service, including link to sources, is available publicly in https://iteavisdom.org/dashboard. The service provides user management and collects together visualizations aimed at different stakeholders. The most relevant visualizations are displayed to each stakeholder by implementing roles for each stakeholder and composing a stakeholder-specific default dashboards from a set of visualizations.

Additionally, Composer allows users to define their own custom dashboards. For instance, a teacher, interested to find out if there are some exercises that caused a lot of difficulties for students, can create a new view that appears here under the view listing in the Composer.

4.2. Utilizing Composer in the Teaching Use Case

In the context of the teaching use case, the Tampere University team has designed and piloted a set of visualizations, which are deployed in the Composer service. When logging in to Composer, users see a different default dashboard, depending on their role. Currently, the default dashboard for teachers consists of a progress view, whereas the default visualization for teaching assistants consists of a status view. Despite the default visualization for students being still work-in-progress, this document introduces the concept of combined EKG and Pulse visualization for the pace of progress.

4.3. Visualizations in the Teaching Use Case

Figure 11 shows the current default dashboard for a teacher. The visualization shown is the Progress view, which shows each student as a line, and how many points each student has received during each course week, since points from weekly exercises and projects map to the final course grade. The teacher can also compare the progress of different student groups to see if there are some behavioral differences by using the group display utilities.

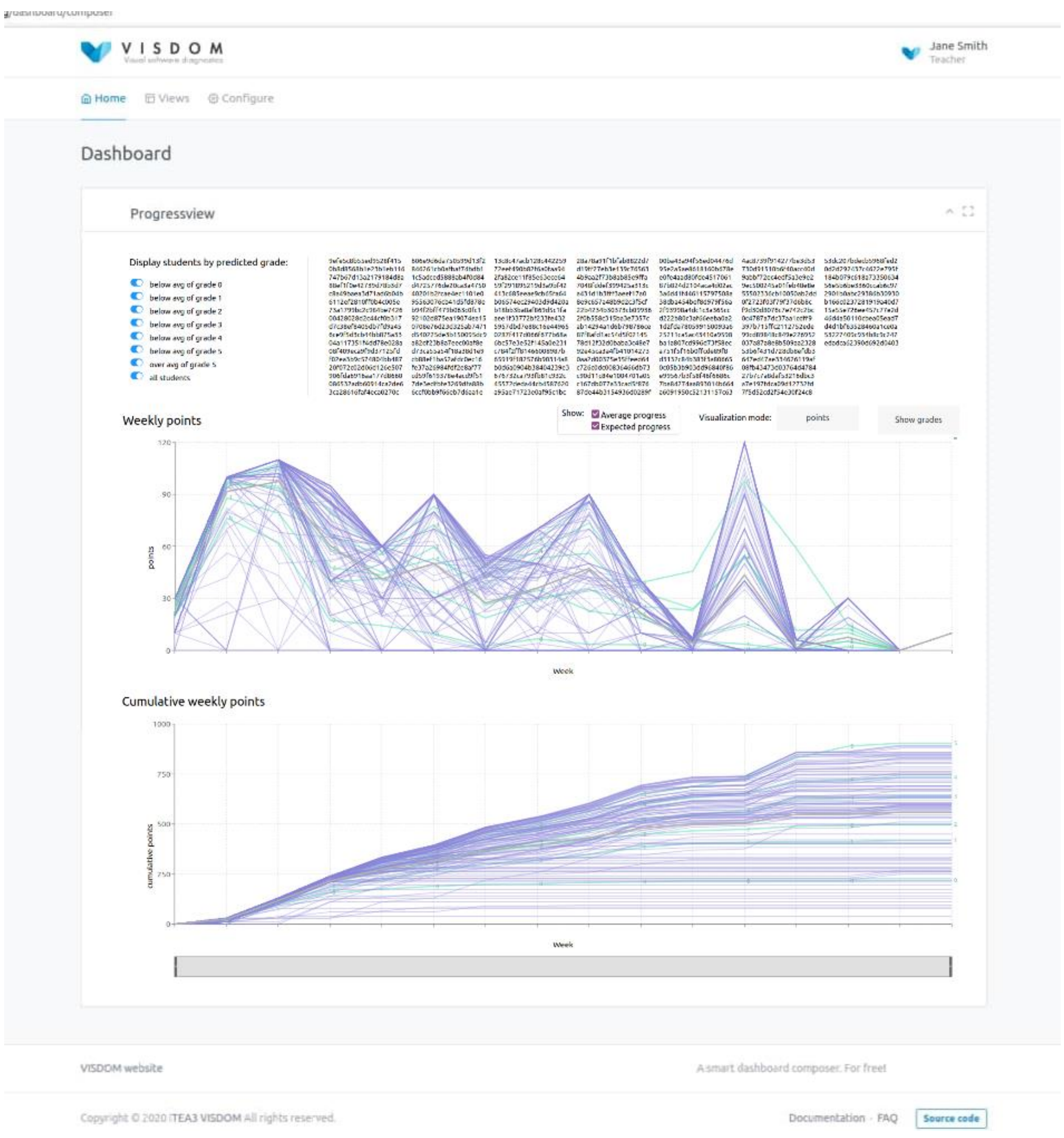


Figure 11: The default dashboard for teachers, showing student progress over the course.

To compare how the default dashboard shows for a different stakeholder, Figure 12 shows the default dashboard for a teaching assistant. We should expect that since the default dashboards should visualize interests of each specific stakeholder group, the dashboard should look different for a user of another role. The visualization seen is the status view, which shows how many points students have collected out of the weekly maximum. Figure 12 shows that on course week 1, there are a few students that have not even started working on the course, but most have successfully completed all weekly exercises.

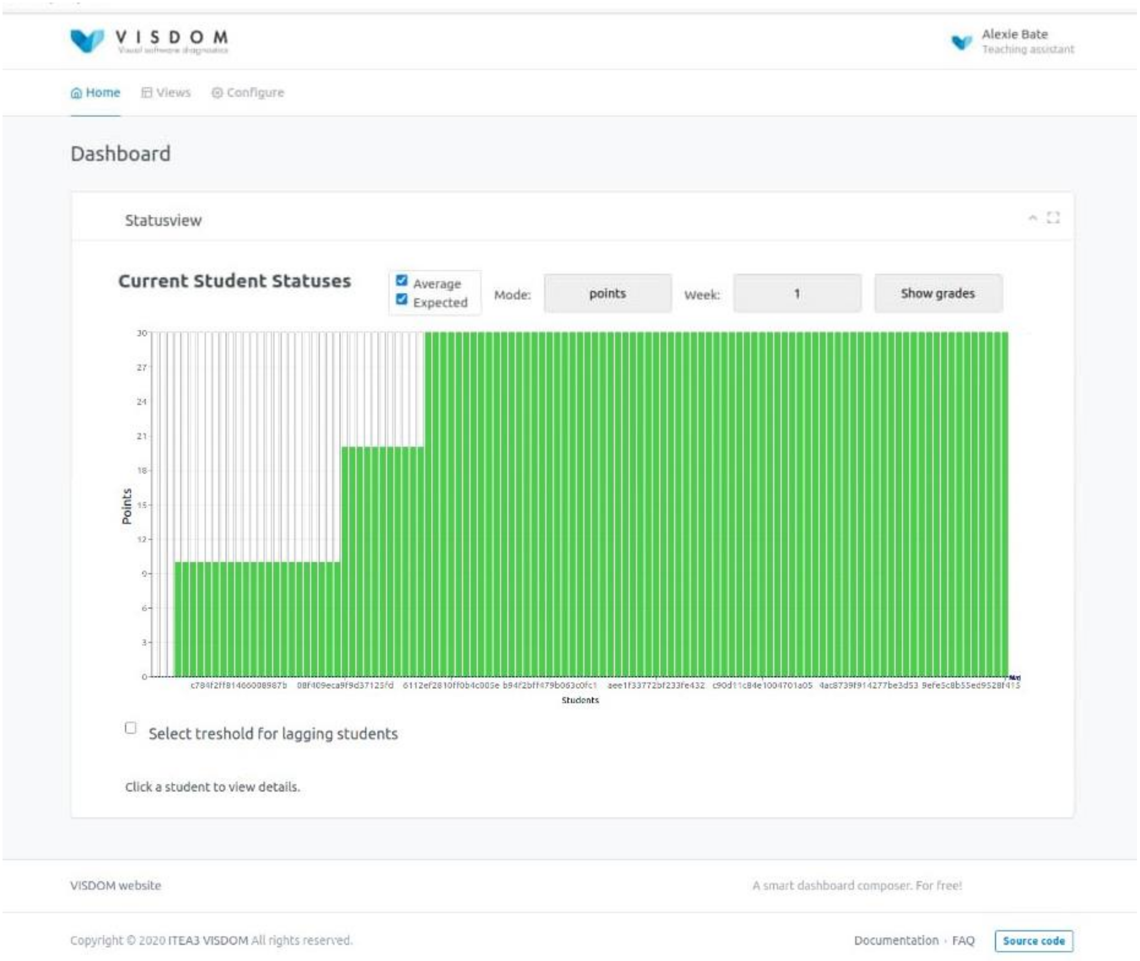


Figure 12: The current default dashboard for teaching assistants showing current status for each student.

For students, we are currently developing a more advanced visualization, combining an EKG and Pulse visualization. The student visualization aims at a proof-of-concept for the advanced visualization principles for EKG and Pulse visualizations, described in deliverable 3.3.1 Designs of new visualizations [1], in which beats in an EKG line translate to the student completing course activities on each course week, showing cyclic progress over the course. The shape of the EKG expresses the estimated workload on each accomplishment. A pulse line constructs of commits made by the student, showing when and how much work the student completes. The pulse line would be shown below the EKG line to allow, for instance, comparing estimated and realized work and their distribution over the course.

4.4. Stakeholder Feedback on the Demo

The different interactions and visual analysis provided by progress and status views are discussed in detail in deliverable 2.3.1 Documented data analysis examples [3]. The analysis examples were also presented to a group of teachers in October 2020 to collect feedback on the implementations. The overall feedback stated that teachers found the system useful, and improvement points were formalized into tickets for future development.

Future development plans for the teaching demo include implementing advanced visualizations. Similarly, the development of the composer service is continued, following the principles described in deliverable 3.2.1 Visualization guidelines [2].

The current configuration of stakeholder specific default dashboards is given manually, but in the future we plan to include a way for the visualizations to communicate in a machine-readable format for whom the visualization is targeted at and what aspects it visualizes. The composer will use that information to generate the default dashboards automatically. Future plans include implementing similar configurations for stakeholders, too, to allow generating default dashboards automatically for new stakeholders.

5. Access to the demos

Quality demo:

As the tool analyses data from an internal project of Experis, which deals with private data, we cannot make the demo or code available to the public. However, a video presenting the functionalities developed until now will be made available on the project's website.

- Running demo: https://visdom-project.github.io/website/.

Teaching demo:

The running demo and source codes are available as follows

- Running demo: https://iteavisdom.org/dashboard
- Source code: https://github.com/visdom-project/visdom

6. Next Steps & Conclusions

The two demos presented in this paper provide two different but complementary approaches to VISDOM´s objective.

The product quality demo dashboard presents data that has been previously analysed, and its visualisations provide information at a glance that supports decision making. The teaching software development demo, on the other hand, focuses on the representation of raw data and allows the design of specific and customised visualisations.

Both approaches are complementary and currently work is ongoing to offer either approach in both use cases.

As an example, the product quality demo foresees the inclusion of a screen named "Raw Data" (see Figure 13) that will offer access to raw data representations. This will allow users to access lower level information enabling them to better understand the rationale behind the indicators represented.

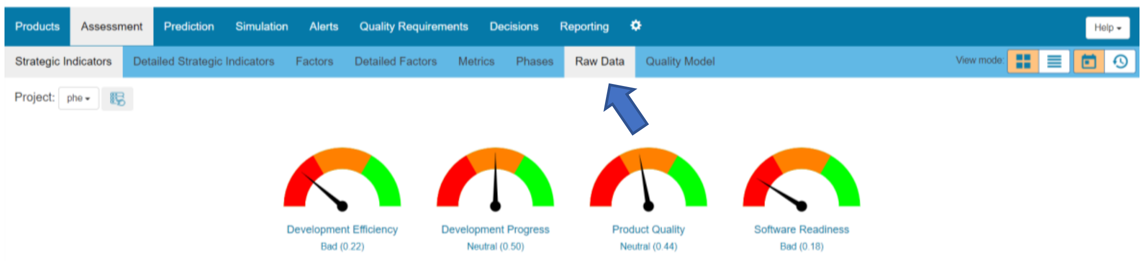


Figure 13: Raw Data Screen for Product Quality Demo

7. References

[1] Designs of new visualizations, VISDOM Deliverable D3.3.1., January 2020. Available through ITEA3 portal, www.itea3.org.

[2] Visualization guidelines for the different viewpoints, VISDOM Deliverable D3.2.1, December 2020. Available through ITEA3 portal, www.itea3.org.

[3] Documented data analysis examples, VISDOM Deliverable D2.3.1. September 2020. Available through ITEA3 portal, www.itea3.org.