2019 HAWAII UNIVERSITY INTERNATIONAL CONFERENCES
SCIENCE, TECHNOLOGY & ENGINEERING, ARTS, MATHEMATICS & EDUCATION JUNE 5 - 7, 2019
HAWAII PRINCE HOTEL WAIKIKI, HONOLULU, HAWAII

CREATING EFFECTIVE AND EFFICIENT USER DASHBOARDS THROUGH DYNAMIC CUSTOMIZATION AND WELL-DESIGNED WEBPAGE VISUALIZATION

CANKAYA, EBRU CELIKEL
ODEKIRK, DYLAN
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF TEXAS AT DALLAS
RICHARDSON, TEXAS

Dr. Ebru Celikel Cankaya
Dr. Dylan Odekirk
Department of Computer Science
University of Texas at Dallas
Richardson, Texas

Creating Effective and Efficient User Dashboards Through Dynamic Customization and Well-Designed Webpage Visualization

Synopsis:

This work presents the design and implementation of a fully functional student dashboard software that promises to facilitate a generic university information system in an efficient and effective way. Our project also serves as a didactic platform as it walks through the steps of developing software from a software engineering perspective. To verify its validity and enhanced features, we compare the performance of our work with similar software available.

Creating Effective and Efficient User Dashboards Through Dynamic Customization and Well-Designed Webpage Visualization

Dylan Odekirk, Ebru Celikel Cankaya

Abstract. This work presents the design and implementation of a fully functional student dashboard software that promises to facilitate a generic university information system in an efficient and effective way. Our project also serves as a didactic platform as it walks through the steps of developing software from a software engineering perspective. To verify its validity and enhanced features, we compare the performance of our work with similar software available.

1. INTRODUCTION

Throughout the past few decades, dashboards have evolved from static visual-reports to interactive user interfaces with different views and purposes [3]. Their usage and implementation varies based on their functionalities and user needs. However, dashboards consistently need to meet two requirements; they need to be effective and they need to be efficient. Dashboards are meant to display various ranges of important data that an individual needs to be able to read, understand, and interpret quickly and easily [1].

2. RELATED WORK

Discussions on dashboard design repeatedly revisit the terms “effectiveness” and “efficiency”, and the importance of implementing these concepts into dashboard design [1]. Before discussing design implementation, we should explore what a dashboard is, how it is defined, and what are its general functionalities. As defined in [1], dashboards are a powerful medium of communication, and, effective dashboards display important data that someone must monitor to do a job in a way that can be read and understood quickly and easily. Another definition emphasizes the importance of timeliness for dashboards: *a predominately visual information, display that people use to rapidly monitor current conditions that require a timely response to fulfill a specific role* [2,3].

3. DEFINING EFFICTIVENESS AND EFFICENCY

The following examples explore the importance of effectiveness and efficiency through variety of existing dashboards: The first example is the Indonesian military vehicle R & D’s dashboard project. This project is intended to optimize system integration through cognitive performance, which needs to effectively display a dashboard interface on military vehicles panels. It employs a design interface that carefully considers situational awareness, interaction with artifacts, and decision making [4]. This dashboard implements these concepts through cognitive clustering. This means that the functionalities of the dashboard are grouped in different clusters that implements the dashboard requirements. For example, one cluster on the panel is a navigation cluster which contains signals, wipers, a washer, and lighting switches. All of these functionalities are similar in functionality and a user can easily infer that functionalities having to do with non-critical vehicle functions can be found in this cluster. If the functionality is critical, like an emergency button, the dashboard would have its own cluster for it. This means there is only one cluster for a critical functionality. From the example of the Indonesian military vehicle panel, we can conclude that through clustering of alike functionalities based on their usage and

immediacy, we will have a more effective and efficient dashboard.

The next example discusses the facilitation of efficient data dissemination for the purpose of monitoring. The School of Public Health at the University of Hong Kong designed an online interactive dashboard for influenza surveillance [5]. This dashboard is specifically for public health surveillance which is defined as, "the ongoing, systematic collection, analysis, and interpretation of health data essential to the planning, implementation, and evaluation of public health practice, closely integrated with the timely dissemination to those who need to know" [5,6]. This means that in order for the dashboard to be effective and efficient, it needs to be "content-rich" and "self-explanatory" when displaying years of disease activity [5]. To fulfill these requirements, the influenza dashboard allows for generous data manipulation in order to keep the dashboard as updated as possible. The dashboard also has multiple pages; a main page, an individual surveillance data stream drill-down page, and a recommendations drill-down page. The main page is meant to be a screenshot of the influenza surveillance, which dynamically displays a clear overview of disease activity at one glance. A well-defined presentation layer is crucial for having an effective and efficient dashboard. In this example, the presentation layer is broken down into a main page with a screenshot of the other more detailed pages. This way both of the fundamental dashboard concepts are implemented: the user can efficiently view everything on the main page, and they can effectively view more detailed pages that are available.

As explored in the previous examples, dashboards need to display information where it is self-explanatory and rich in content. And as we discussed, this can be achieved through cognitively clustering different functionalities and by displaying disseminated data with the use of multiple pages e.g. a main page.

In the last example, we emphasize a new concept which contributes to the effectives and efficiency of dashboard design. This concept is dynamic customization. Dashboards are most commonly implemented as a webpage or some kind of graphical interface displayed on a monitor, mobile phone, etc., which allow for multiple layers of data integration, collection, and overall manipulation. An effective and efficient dashboard needs to be dynamic and customizable on command. Authors in [7, 8] defines this as a self-regulating dashboard, where it allows for easy navigation to more complete information on analysis views. Self-regulation can be achieved through different levels of customization: low-level, where users are able to set basic parameters; medium-level, where users can define different kinds of calculation functions and alter the visuals; and high-level, where learners are able to manipulate traces used in the dashboard [7]. Depending on the platform the dashboard is being utilized e.g. webpage, mobile etc., it is important to note that the higher the customization level is, the higher the computer background requirements are.

4. EXPERIMENTAL WORK

In this section, we describe our experimental implementation of an effective and efficient dashboard. Our concept is implemented as a student dashboard.

4.1. DESIGN

We adopt a software engineering approach to design and implement our project. In this section, the chronologic steps of this process is explained in detail.

In designing the Student Dashboard software, we start by collecting the functional and non-functional requirements. They help us determine the functionality of the pages that needs to be implemented. The requirements are as follows:

Overall System Functional Requirements

1. The user should be able to log-in to a portal
2. The dashboard should display five different tabs listed as:
    a. Home
    b. Class Registration
    c. Calendar
    d. Homework
    e. Billing management

Home Page Functional Requirements

1. The user should be able to view their grades.
2. The user should be able to see the weather in their desired location.
3. Able to see upcoming homework that is due.
4. Should see the stock market.
5. User should see all bills that are due.
6. All of the above should be in the form of customizable widgets.

Class Registration Functional Requirements

1. The user should be able to register for classes.
2. The user should be able to view available courses.
3. The user should be able to view the course catalog.

Custom Calendar Functional Requirements

1. The user should be able to create different calendars.
2. The user should be able to add events to the calendar and customize by date, time, frequency, and color coordinate the event.

Homework Page Functional Requirements

1. The user should be able to view upcoming homework assignments due for each week.
2. The user should be able to add homework and their due dates.
3. The homework page should give the user reminders when the upcoming due dates are approaching.

Billing Management Functional Requirements

1. The user can view their bills and holds.
2. The user can purchase their parking permit.
3. The Billing page should display any grants, scholarships, or other financial aid that has been offered or received.

Non-Functional Requirements

1. The dashboard shall be able to access the university coursebook.
2. Users of the dashboard shall be able to access the material from eLearning.
3. The dashboard shall be able to implement a third party weather widget for the home page.

4. The user shall be able to implement a stocks widget for the homepage.
5. The dashboard shall be able to access a grade portal for the user's grades.

The analysis of functional and non-functional requirements lead us to create the use case diagram presented in Fig 1.

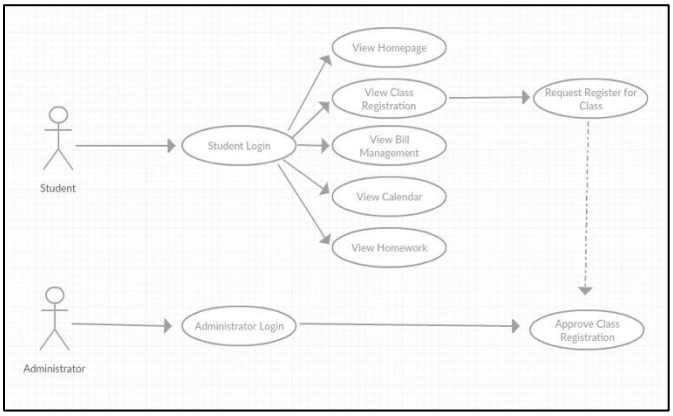

Figure 1. Use Case Diagram for the Student Dashboard Software

Once the basic tabs for the dashboard page, we determined the functionality of the home page, as seen in Figure 2.

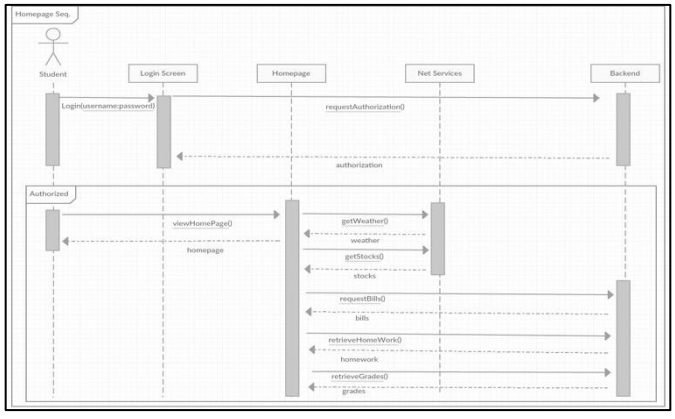

Figure 2. Sequence Diagram for the Home Page of the Student Dashboard Software

The dashboard implemented React/ Redux for a view controller. We appropriately separated the presentation and the interaction from the system data. The dashboard is the presentation and interaction from the user, while the system data is the functionality of the dashboard. The view component sends out user events to the controller via React and Redux as a user interface. Our architectural design therefore followed that of a Model-View-Controller, which is illustrated in Figure 3.

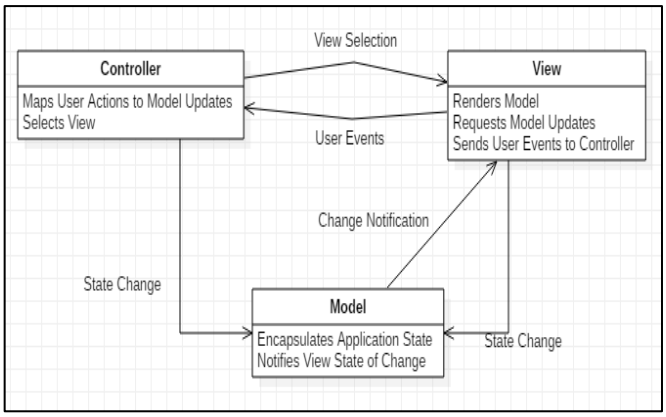

Figure 3. Architectural Design using MVC for the Student Dashboard Software

Lastly, Figure 4 illustrates the student dashboard software class diagram.

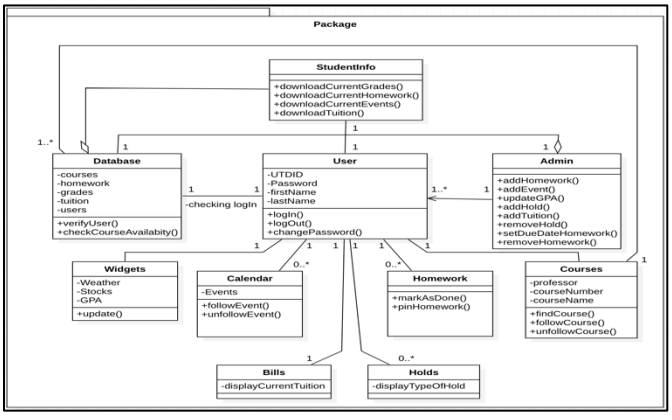

Figure 4. Class Diagram for the Student Dashboard Software

5. IMPLEMENTATION

Our physical implementation of the student dashboard is in the form of a webpage. The webpage utilizes a JavaScript library called React and Redux. React is defined as a JavaScript library for building user interfaces. It designs simple views for each state in the application, and React will efficiently update and render just the right components when the data changes [9].

The React and Redux library allows us to dynamically render React components through a render() method that takes input data and returns what information to display [9]. This allows for more liberal data manipulation and customization on a high-level [7].

React is also considered component-based. This means to be able to build encapsulated components that manage their own state, then compose them to make complex UIs [9]. These features help us create the self-regulating dashboard as discussed in the Related Work section above [8].

Figure 5 shows the main page of our student dashboard. Notice that we have three sections at the top, which illustrates cognitive functionality clustering as discussed previously. The sections at the top are grouped in similar ways to display their necessary information with ease. The sections below include an open-source weather widget and an "Announcements" section. Both of these components are dynamic, and change as external sources manipulate data that ultimately refresh the dashboard for the user to see. Data entry is accessible for students, where they can enter data manually and for instructors that can enter data or announcements on their end.

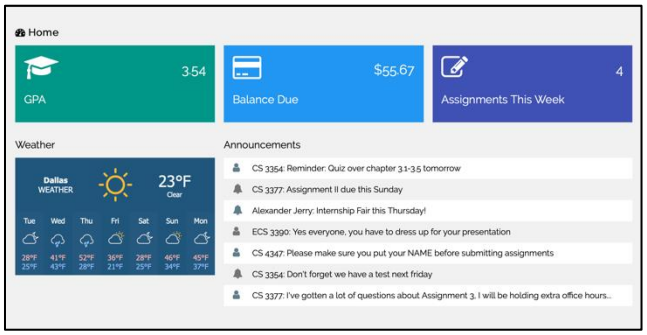

Figure 5. Main Page of Student Dashboard

5.1. COMPARISON WITH SIMILAR WORK

From our previous discussion on defining an effective and efficient dashboard, we defined specific parameters to physically measure the effectiveness and efficiency of a dashboard. These parameters are data visualization, front-end implantation, customization options [10], query implementation, and maintainability.

Using these five parameters, we compare our dashboard with three other existing dashboards used for various purposes. However, keep in mind that these parameters should apply to most dashboards despite their specific usage.

The first dashboard is for small-business accounting management, it is an existing service called Quickbooks which maintains these services [11] as seen in Figure 6.

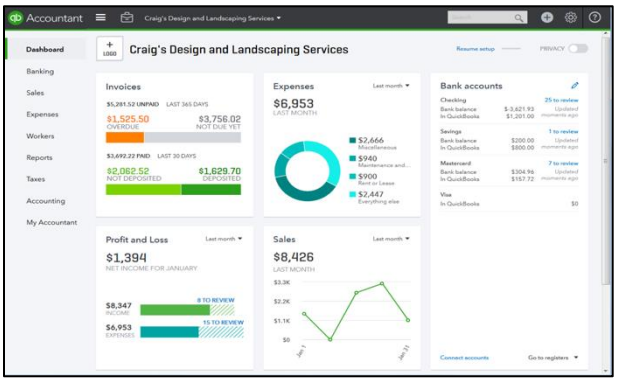

Figure 6. Quickbooks Dashboard [11]

The second dashboard is an open-source dashboard which implements React-Material [12] which is seen in Figure 7.

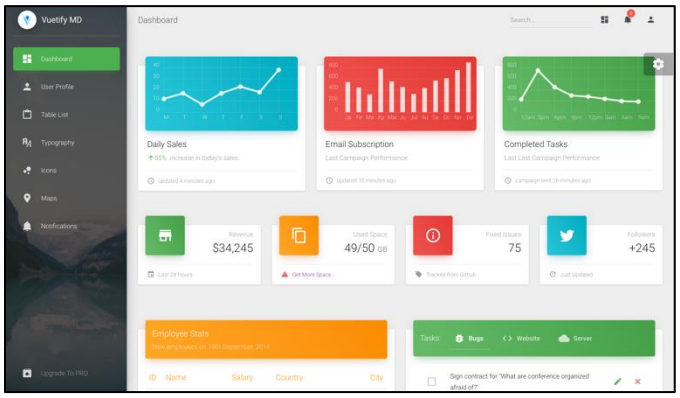

Figure 7. React/ Redux Dashboard [12]

The last dashboard we used for comparison purposes is a mock-up of a student dashboard from BrainCert Academy [13] as seen in Figure 8.

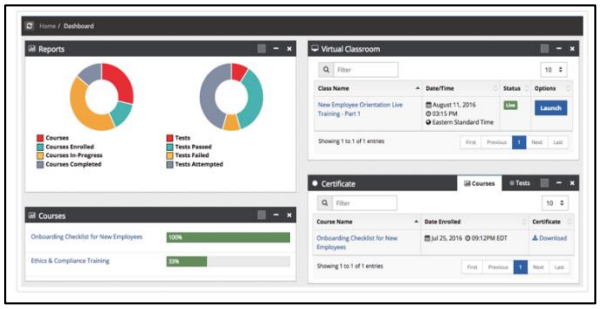

Figure 8. Student Dashboard [13]

An overall comparison view w.r.t. the five parameters is presented in Table 1 below. The table is read as follows:
Q.B: Refers to the Quickbooks Dashboard in Figure 6
R/R: Refers to the React/Redux Dashboard in Figure 7
SD: Refers to the Student Dashboard in Figure 8
Ours: Refers to our dashboard in Figure 5

On the vertical side, the numbers represent the parameters discussed earlier. If there is a check, the corresponding dashboard contains these parameters:
1: Data Visualization
2: Front-end implantation
3: Customization options
4: Query implementation
5: Maintainability

Table 1. Table of dashboards with parameter comparison.

	Q.B	R/R	SD	Ours
1	✓	✓	✓	✓
2	✓	✓	✓	✓
3				✓
4				✓
5	✓			✓

6. CONCLUSION

We present the design and implementation of a student dashboard software that promises to provide its features in an efficient and effective way. Adopting a software engineering approach, this work also serves to the purpose of demonstrating a didactic project that walks through the steps of this engineering methodology to developing the software.

As part of future work, we plan on deploying our implementation for test use so as to collect user feedback. This will allow us to improve our design by regarding user requests and further requirements. We will also include more a larger span of similar work for comparison purposes so as to see where our design resides among others.

ACKNOWLEDGEMENT

The authors would like to thank the students, Jace Baker, Joyce Choo, Michael Egar, Michel Mazal, Code Sachse, and Will Spencer for their contribution to the initial design of this software.

REFERENCES

[1] S. Few, "Intelligent Dashboard Design," DM Review, vol. 15, (9), pp. 12, 2005. Available: http://libproxy.utdallas.edu/login?url=https://search.proquest.com/docview/214671399?accountid=7120.

[2] S. Few, Blog post: *There's nothing mere about semantics*, 2017, [online] Available: https://www.perceptualedge.com/blog/?=2793

[3] A. Sarikaya, M. Correll, L. Bartram, M. Troy, and D. Fisher. (2019). *What Do We Talk About When We Talk About Dashboards? - IEEE Journals & Magazine*. [online] Ieeexplore.ieee.org. Available at: https://ieeexplore.ieee.org/abstract/document/8443395 [Accessed 17 Jan. 2019].

[4] B. Iqbal, A. Suzianti, and B. Nurtjahyo. "Military Vehicle Dashboard Design Using Semantics Method in Cognitive Ergonomics Framework." Vol. 9174. Springer Verlag, 2015. 152–163.

[5] C. Cheng, D. KM, B. Cowling, L. Ho, G. Leung, and E. Lau. (2011). "Digital Dashboard Design Using Multiple Data Streams for Disease Surveillance with Influenza Surveillance as an Example." *Journal of Medical Internet Research*. [online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3222192/

[6] S. Thacker. Historical development. In: Lee LM, Teutsch SM, Thacker SB, St Louis ME, editors. Principles and Practice of Public Health Surveillance. 3rd edition. New York, NY: Oxford University Press; 2010

[7] M. Ji, C. Michael, E. Lavoue, and S. George. (2014) "DDART, a Dynamic Dashboard for Collection, Analysis and Visualization of Activity and Reporting Traces." *Open Learning and Teaching in Education Communities.* [online].

[8] Michel, C., Lavoué, E.: KM and Web 2.0 Methods for Project-Based Learning. MEShaT: a Monitoring and Experience Sharing Tool. Multiple Perspectives on Problem Solving and Learning in the Digital Age. p. 49-66. Ifenthaler D., Isaias P., Spector J.M., Kinshuk, Sampson D., Heidelberg (2011).

[9] Facebook Open Source, "React: A JavaScript Library for Building User Interfaces" (2018). [Online] Available: https://reactjs.org/

[10] Orts, D. "Dashboard design and implementation: a step-by-step guide." (2007) *KMWorld*, *16* (1) [Online].

[11] Creative Tim. "Premium Bootstrap Themes." (2018). [Online] Available: https://www.creative-tim.com

[12] Intuit Quickbooks. "How it Works" (2018). [Online] Available: https://quickbooks.intuit.com/how-it-works/

[13] BrainCert Academy. "Introducing New Student Dashboard for Enterprise LMS", (2016) [Online]. Available: https://www.braincert.com/blogs/17learning-management-system/142-introducing-students-dashboard-for-enterprise-lms