BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

VISUALIZATION OF DATA FROM RACING SPECIALS USING DASHBOARD-TYPE SCREENS

VIZUALIZACE DAT ZE ZÁVODNÍCH SPECIÁLŮ POMOCÍ OBRAZOVEK TYPU DASHBOARDŮ

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **JAKUB KAČKA**
AUTOR PRÁCE

SUPERVISOR **Ing. PETR JOHN**
VEDOUCÍ PRÁCE

BRNO 2025

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Bachelor's Thesis Assignment

164749

Institut:	Department of Information Systems (DIFS)
Student:	**Kačka Jakub**
Programme:	Information Technology
Title:	**Visualization of Data from Race Specials Using Dashboards**
Category:	Information Systems
Academic year:	2024/25

Assignment:

1. Study the Internet of Things (IoT). Primarily focus on its use in the automotive industry.
2. Study the principles of data visualization on dashboard screens and key performance indicators (KPIs).
3. Analyze the requirements for dashboard screens placed in racing specials. Focus primarily on off-road trucks. Consult with specialists, mechanics, and racers if possible.
4. Based on the analysis, design possible visualizations and a system providing comprehensive dashboard screens.
5. After consultation with the supervisor, implement the designed system.
6. Evaluate the chosen approach and testing with target users.

Literature:

- Greengard, S.: The Internet of Things. MIT Press, 2015, ISBN 978-026-2527-736.
- John, P. Optimising processes in IoT. Brno, 2024. PhD thesis proposal. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Ing. Tomáš Hruška, CSc.
- Few, S.: Information Dashboard Design: The Effective Visual Communication of Data. Sebastopol, USA: O'Reilly, 2006, ISBN 978-059-6100-16

Requirements for the semestral defence:

- Points 1 - 4.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**John Petr, Ing.**
Head of Department:	Kolář Dušan, doc. Dr. Ing.
Beginning of work:	1.11.2024
Submission deadline:	14.5.2025
Approval date:	22.10.2024

Abstract

Motorsport demands that vehicles deliver maximum performance even under extreme conditions, requiring mechanics to continuously monitor the vehicle's status and make rapid decisions. This thesis presents a functional software prototype of a digital dashboard designed specifically for the onboard mechanic in a rally vehicle. Implemented as a web application with a React frontend and Node.js backend which is responsible for the simulation. A reusable library of custom visualization components was developed to support future dashboard implementations. The solution was tested in collaboration with industry professionals, resulting in a validated design tailored for motorsport use, while also demonstrating generalizable principles for rapid and reliable sensor data visualization in other domains.

Abstrakt

Motorsport vyžaduje, aby vozidlá dosahovali maximálny výkon aj v extrémnych podmienkach. K tomu je potrebné, aby mechanici mohli počas celých pretekov monitorovať aktuálny stav vozidla a robiť rýchle rozhodnutia. Výstupom tejto práce je funkčný softvérový prototyp digitálneho dashboardu navrhnutý špeciálne pre palubného mechanika v rally vozidle. Dashboard bol implementovaný ako webová aplikácia s frontendom v Reacte a backendom v Node.js ktorý je zodpovedný za beh simulácie. Súčasťou je vlastná knižnica vizualizačných komponentov ktorá môže byť využitá pri vývoji dalších podobných dashboardov. Riešenie bolo testované s odborníkmi z praxe. Práca tak prináša nielen overený návrh a implementáciu pre konkrétne použitie v motoršporte, ale aj univerzálne princípy využiteľné v iných oblastiach, kde je potrebná rýchla a spoľahlivá vizualizácia senzorových údajov.

Keywords

Motorsport Telemetry, Dashboard Visualization, React Web Application, Sensor Data Monitoring, Internet of Things (IoT), Real-Time Error Detection.

Kľúčové slová

Telemetria v motoršporte, Dashboardové vizualizácie, Webová aplikácia v React, Monitorovanie dát zo senzorov, Internet of Things (IoT), detekcia chýb v reálnom čase.

Reference

KAČKA, Jakub. *Visualization of data from racing specials using dashboard-type screens.* Brno, 2025. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Petr John

Rozšírený abstrakt

Rallye Dakar patrí medzi najnáročnejšie a najextrémnejšie motoristické podujatia na svete. Ide o preteky, ktoré kladú obrovské nároky nielen na pilotov, ale predovšetkým na technické vybavenie a celkovú spoľahlivosť vozidiel. Vzhľadom na dĺžku tratí, klimatické extrémy a vysokú mieru opotrebenia je kľúčové, aby bolo vozidlo neustále monitorované a aby mechanik počas jazdy dokázal reagovať na meniaci sa technický stav vozidla. Zatiaľ čo vodič sleduje predovšetkým jazdné parametre ako rýchlosť či otáčky, palubný mechanik musí sledovať širšie spektrum diagnostických údajov týkajúcich sa stavu kritických systémov. Práve preto vznikla potreba navrhnúť špecializovaný digitálny dashboard, ktorý bude určený výhradne pre mechanika v nákladnom vozidle, a to s dôrazom na maximálnu čitateľnosť, efektivitu vizualizácie a rýchlosť interpretácie zobrazovaných hodnôt.

Práca v prvej časti skúma koncept Internetu vecí (IoT), ktorý dnes presahuje rámec individuálnych smart zariadení a zahŕňa komplexné systémy schopné zberu, prenosu a vyhodnocovania údajov v reálnom čase. IoT predstavuje ekosystém prepojených objektov – od senzorov cez mikrokontroléry až po cloudové služby. V automobilovom priemysle a najmä v motorsporte tento koncept nadobúda nový rozmer. Pretekárske špeciály sú v podstate pohyblivé IoT uzly, ktoré v reálnom čase generujú dáta o rôznych častiach vozidla. Ich správna interpretácia je nevyhnutná na predchádzanie poruchám a optimalizáciu výkonu vozidla. V rámci tejto práce bolo analyzované, ako sa dáta z týchto senzorov spracúvajú, ako sa prenášajú v rámci CAN zbernice a aké výzvy prináša ich vizualizácia v prostredí s vysokým časovým stresom a zhoršenou viditeľnosťou.

Dôležitou súčasťou návrhu bolo aj detailné oboznámenie sa s princípmi vizualizácie dát. Efektívna vizualizácia je základom rýchlej orientácie a správneho rozhodovania. V práci boli preto podrobne preskúmané existujúce vizualizačné techniky a princípy, ako sú preattentívne vnímanie, Gestalt princípy (blízkosť, podobnosť, kontinuita, uzatvorenie) či pravidlá rozloženia informácií na obrazovke podľa vizuálnej dôležitosti. Tieto princípy boli následne aplikované v návrhu dashboardu, pričom boli použité vizualizačné médiá ako bullet grafy, nelineárne bary, trendové šípky či farebné indikátory. Znalosť týchto techník bola kľúčová pri návrhu riešenia, ktoré bolo následne konzultované s odborníkmi z praxe, najmä s Ing. Davidom Svídom, PhD., ktorý má bohaté skúsenosti s vývojom motorov pre vozidlá Rallye Dakar.

Na základe teoretického základu a odbornej spätnej väzby boli navrhnuté dve verzie dashboardu – prvá vychádzala z poznatkov literatúry a princípov efektívnej vizualizácie, druhá bola upravená po konzultáciách s odborníkmi. Finálny návrh je rozdelený do funkčných blokov – zobrazenie kľúčových komponentov (tlaky a teploty pneumatík), diagnostické informačné dlaždice („info tiles“) a systém chybových hlásení. Dôležitým výstupom návrhu je vývoj vlastných vizualizačných komponentov, ktoré umožňujú rýchlu a efektívnu interpretáciu technických údajov. Medzi kľúčové prvky patrí nelineárny bar, ktorý zvýrazňuje dôležité intervaly meraných hodnôt, trendová šípka zobrazujúca vývoj veličiny v čase na základe kĺzavého priemeru a historický graf, ktorý umožňuje spätne analyzovať správanie konkrétneho senzora počas uplynulých minút.

Celý systém bol implementovaný ako webová aplikácia s použitím React (frontend) a Node.js s WebSocket serverom (backend). Dashboard umožňuje načítanie CSV súborov so záznamami zo senzorov a prehrávať ich ako simuláciu v reálnom čase. Tento prístup umožnil testovanie prototypu bez nutnosti napojenia na reálnu senzorovú sústavu vozidla. Dáta sú následne vizualizované v komponentoch dashboardu.

V rámci testovania bola aplikácia predstavená štyrom odborníkom s priamou skúsenosťou z Rallye Dakar. Patrili medzi nich mechanici sediaci priamo vo vozidle počas etáp, ako aj

servisní technici zodpovední za údržbu medzi etapami. Ich spätná väzba potvrdila vysokú mieru prehľadnosti navrhnutého riešenia a prínos vybraných vizualizačných techník. Oceňovali najmä zreteľné farby, interaktívne hlásenia a konzistentnosť zobrazenia. Na základe ich pripomienok boli navrhnuté viaceré rozšírenia, ako napríklad možnosť blikajúceho pozadia pri výskyte chyby, použitie externého tlačidla na potvrdenie chýb alebo úplné vypnutie chybných senzorov, ktoré by inak spôsobovali zahltenie používateľa zbytočnými upozorneniami.

Výsledkom práce je plne funkčný prototyp digitálneho dashboardu určeného pre nasadenie v extrémnych podmienkach rallye. Navrhnutý systém je modulárny a prenosný, čo umožňuje jeho adaptáciu aj v iných oblastiach motorsportu či automobilovej diagnostiky, kde je potrebná rýchla a spoľahlivá interpretácia technických údajov. Navyše, použitá architektúra systému umožňuje jednoduchú integráciu ďalších senzorov, konfiguráciu vizualizácií podľa potrieb používateľa a budúce rozšírenie o reálne vstupy z CAN zbernice. Práca tak predstavuje dôležitý krok smerom k inteligentnejšiemu monitorovaniu vozidiel v reálnom čase, založenému na overených princípoch vizualizácie a praktických skúsenostiach z terénu.

Visualization of data from racing specials using dashboard-type screens

Declaration

I declare that I have prepared this bachelor thesis independently under the supervision of Mr. Ing. Petr John. Further information was provided by Ing. David Svída Ph.D. and Ing. Martin Beran. I have listed all literary sources, publications and other sources from which I have drawn.

.....
Jakub Kačka
May 13, 2025

Acknowledgements

I would like to thank my thesis supervisor Ing. Petr John for his patience in solving all the problems associated with this work. Furthermore, I would like to thank Ing. David Svída, Phd. and Ing. Martin Beran for providing me with valuable information in the field of Dakar Rally racing and also for consulting my solutions. I would like to thank my partner for her support throughout the process of making this thesis. And last but not least, I would like to thank my parents without whom the whole study that led to this thesis would not have been possible.

Contents

List of Figures

Chapter 1

Introduction

Motorsport, particularly rally racing, places extreme demands on vehicles and crews. In events like the Dakar Rally, vehicles endure harsh environments: heat, dust, vibration, and long stages with limited service opportunities. In such conditions, real-time monitoring of the vehicle's condition is critical—not only to optimize performance but to detect and react to technical issues before they become race-ending failures. The onboard mechanic must interpret a wide range of diagnostic data and respond immediately. The efficiency of this process heavily depends on how clearly and intuitively the information is displayed.

This thesis presents the design and implementation of a digital dashboard tailored specifically for the onboard mechanic in a rally truck. Unlike conventional dashboards, often optimized for the driver or generalized for multiple roles, this solution prioritizes visual clarity, perceptual speed, and context-aware display of information. The goal is to reduce reaction time and cognitive load through effective visualization techniques, enabling the mechanic to focus on what matters most.

The theoretical basis of this work lies in two main areas: the Internet of Things (IoT), which defines the network of sensors embedded in modern vehicles, and data visualization principles, which guide how information should be displayed for clarity and usability. Concepts such as preattentive processing, layout design, and key performance indicators (KPIs) are applied to support fast and accurate decision-making.

To validate and improve the design, several commercial dashboard systems were analyzed, exposing limitations in usability and diagnostic depth. Two versions of the dashboard were proposed: an initial version based on theory and a revised version refined through expert feedback from Dakar Rally engineers. The final prototype was implemented as a web application using React and Node.js, featuring real-time CSV simulation, nonlinear bar graphs, modular info tiles, trend indicators, and interactive error messaging.

The solution was tested with experienced Dakar participants, whose feedback confirmed its clarity and practical value. Suggestions such as blinking alerts and sensor deactivation are noted as possible future enhancements. This work ultimately delivers a robust and extensible dashboard for high-pressure motorsport environments.

Chapter 2 explores the IoT framework with emphasis on its role in motorsport. Chapter 3 introduces data visualization principles relevant to dashboards. Chapter 4 defines system requirements and evaluates existing solutions. Chapter 5 presents the proposed designs, followed by implementation details in Chapter 6. Chapter 7 summarizes user testing and feedback. Chapter 8 concludes the thesis and outlines future improvements.

Chapter 2

Internet of Things

Internet of Things (IoT for short) is a term first used by Kevin Ashton in 1999, when he refered to IoT as a uniquely identifiable interoperable connected objects with radio-frequency identification (RFID) technology [16, 18]. But the history of IoT-like devices started a long time before this. In the 1980s, programmers at Carnegie Mellon University made the first Internet appliance, the smart Coke machine. Programmers working several floors above the machine wrote a server program that was tracking the time of the coke being stored in a fridge, so they knew if it was worth it to go down the floors and the coke they wanted to buy was already cold [18].

The Internet of Things has many definitions depending on the implementation technology. Objects in IoT can be identified uniquely in the virtual representation, and all things within IoT are able to exchange data and, if needed, process the provided data depending on the usage [16]. So the IoT is a big network of things connected, wirelessly or by wire, to the internet or between each other.

The first thing that comes to someone's mind who doesn't really deep dive into the problematics of IoT is a smart home. Using a bunch of sensors and smart devices to control your house from your phone sounds futuristic and is probably a great example of how the IoT works and how it can make your life easier [12]. There are a lot of things that you can automate in your house. For example, you have the coffee machine make you a cup of coffee every morning before work at the same time, so when you come to the kitchen, it's waiting for you, and you don't have to waste time with such a boring activity [12]. You can have automatic lights that switch on and off based on your presence in the room, a smart doorbell that can alert you in your mobile app if someone came to your front door, smart thermostat so you can set your temperature to vacation mode when you go away for the weekend and so many other things that can help you with everyday activities and save you some time and money. [12].

However, the smart home is not the only field in which IoT is used. We can also find IoT devices in logistics and supply chain management for tracking information, location, and stock of goods [34]. In healthcare, we can use IoT for constant monitoring and control of health parameters, even if the patient is not present in the hospital, so the system will send information to the healthcare center if any unwanted change in the parameters happens [34]. In transportation, IoT is used for road condition monitoring, license plate identification, remote performance monitoring [34], and much more, which will be discussed in Sections 2.3 and 2.4. Although this is just a small part of the IoT used in the world, hopefully, it shows what it's capable of. [34]

2.1 Types of IoT devices

The main part of IoT networks is objects. All of the sensors, actuators, and any smart devices are objects. That's why the authors of [11] have decided to divide these groups into two categories: Smart and Non-smart objects. Their definition of Smart Objects states: "A Smart Object, also known as an Intelligent Product, is a physical element that can be identified throughout its life and interact with the environment and other objects." And the Non-smart objects consist of sensors and actuators [11].

Sensors are an essential part of the IoT components. Its role is to sense the different environmental changes [21]. Sensors can sense different parameters based on their usage, such as light fluctuation using a photoresistor, temperature using the thermistor, and detecting flames, sounds, movements, or anything else [11]. There are many different categories to group sensors by; however, Mohamed in [21] has divided them into three main categories, mechanical, electrical, and chemical sensors, based on the technology of the sensor. There can be analog or digital sensors based on the output of the sensors. They are also divided into active or passive sensors.

As sensors are the main part of the IoT network, some requirements include accuracy, resolution, and sensitivity. The whole functionality of the systems is based on these parameters, as the sensors provide feedback to the controls that provide signals to the actuators [21].

Actuators are the part of the IoT systems that perform actions based on the readings of the sensors [21]. Actuators come in three main types: mechanical, electrical, and pneumatic. They are responsible for transforming energy into movement. Essentially, an actuator is a device that changes an electrical input into mechanical action or another practical form of energy [21].

Smart objects usually have an embedded operating system and actuators, sensors, or both, which is why smart objects can communicate with other objects, process data from sensors, and trigger events [11]. However, this is not the only definition of Smart Objects that has been provided. In [33], they consider Smart objects as Intelligent products that constantly monitor, react, and adapt to the environment and can provide optimum performance and active communication.

Mountrouidou [23] provides a taxonomy with a more overall look at the IoT devices that comprise the whole network. The main building blocks of the Network of Things, which is referred to as the Internet of Things, are sensors, aggregators, communication channels, eUtilities, and decision triggers. Sensors have the same definitions as mentioned before. Aggregators are software that can process data. Communication channels are media through which the data flows. eUtilities are products, either hardware or software, for example, a phone or a cloud. Decision Triggers are predicates that must be true if something should happen [23]. Based on these building blocks, the decision tree taxonomy, which can be seen in Figure 2.1, has been created.

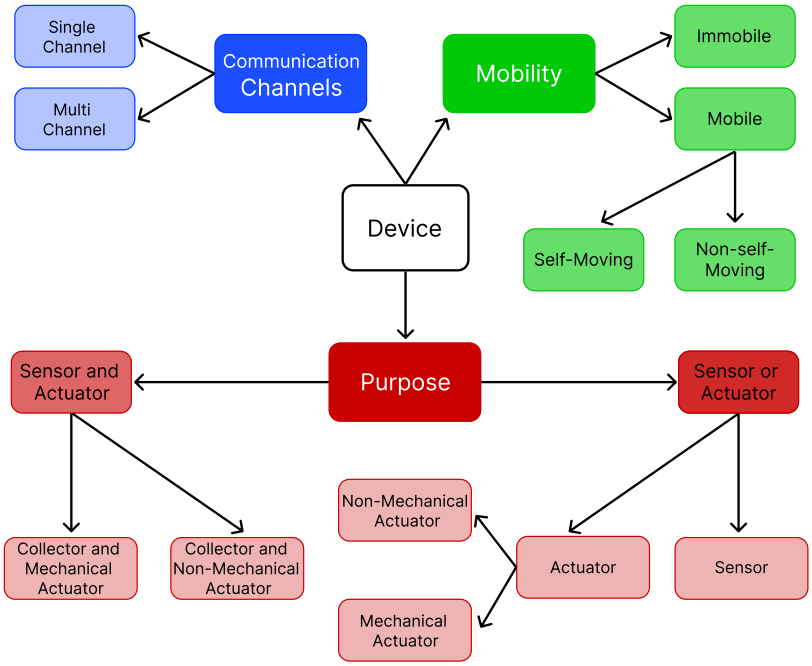


Figure 2.1: Taxonomy tree provided by Montrouidou in [23]

The tree's root represents any device, with the first level outlining general features applicable to all IoT devices [23]. These features align with the NIST primitives. Specifically, the attributes Purpose and Mobility link to actuators; Communication Channels correspond to communication types; and Purpose connects to eUtilities, aggregators, decision triggers, and sensors. Each characteristic at the first level is the root of a binary decision sub-tree, which helps classify devices according to that feature. The subsequent levels in the decision tree feature binary divisions, representing contrasting branches and end points. These lower levels group IoT devices into distinct, complementary categories.

2.2 Communication

In the world of the Internet of Things (IoT), devices must communicate with each other and with central systems to exchange information. Depending on the application, different communication methods are used, balancing factors such as range, energy consumption, data transfer rate, cost, and reliability.

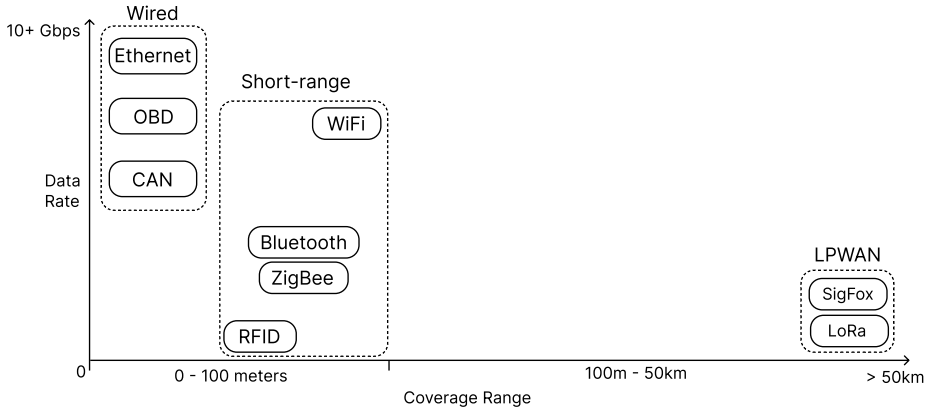


Figure 2.2: Visual representation of the range and data rate of different communication types.

LPWAN technologies

Low power wide area network (LPWAN) is a connection technology used in IoT where the high data rate and low latency aren't the main focus of the system. LPWAN has different key performances such as energy efficiency, scalability, and coverage [30].

LoRa (Long Range) is a physical layer protocol based on LPWAN that uses a spread spectrum technique called Chirp Spread Spectrum (CSS) [30]. CSS spreads the signal over a wide range of frequencies to achieve high interference resilience and also reduce the signal-to-noise-and-interference ratio required for correct data decoding by the receiver. LoRa also has a spreading factor (SF), which can be set from seven to twelve to make it possible to provide data rates and tradeoffs between throughput and coverage range, link robustness, or energy consumption [30]. Higher SF provides a longer range but slower data, while lower SF provides faster data rates but a shorter range.

LoRaWAN is a LoRa-based communication protocol that manages communication between devices, gateways, and the network server. Getaways relay messages between the end devices and the central network server. Devices used in LoRaWAN are defined in three groups. Group A devices are more focused on battery saving, initiating communication by sending data, and only using downlink (communication heading to the device) communication for a short amount of time. Devices in Group B are more commonly used for two-way communication. Group C devices have their receive slots open all the time except when they need to transmit [6, 2].

Another low-power wide area network technology is **SigFox**, which uses ultra-narrowband to provide complete end-to-end connectivity. SigFox uses an Ultra-Narrow Band Modulation to minimize power consumption and allow the signals to travel long distances in rural areas, up to 30-50 km, and in urban locations, three to ten km. SigFox has its own network infrastructure, which is similar to that of a cellular network. However, some of the LoRaWan gateways have been compatible with SigFox since this technology's opening to the public[1] last year. Devices connect to SigFox base stations, which relay data to the cloud.

The SigFox network is built to support 12 bytes of packet size for each message going uplink; the number of messages per day is restricted to 140. Every message going downlink only supports eight bytes of packet size for each message, and the number of downlink mes-

[1] https://www.sigfox.com/unabiz-opens-sigfox-0g-technology-device-library-to-drive-technology-convergence-and-massive-iot/

sages is restricted to only four a day. This indicates that SigFox was initially invented only to support uplink communication, but later, the possibility of downlinking was introduced [6, 2].

Short-Range

Short-range communication technologies are mainly used to support connectivity within a small coverage area [6]. That's why these technologies are best-fit for consumer use-case, rather than big industrial usage [2].

Bluetooth, which was standardized by the Institute of Electrical and Electronics Engineers (IEEE) 802.15.1, is a "wireless technology primarily used for communications between portable devices distributed in a small area (maximum of 100m)" [6]. Bluetooth operates mainly in the 2.4GHz ISM band with data rates up to 3Mbps. Because of Bluetooth's high energy consumption, Bluetooth Low Energy (BLE) was introduced in Bluetooth 4.0 specifically for low-powered IoT devices. Unlike classic Bluetooth, which uses a continuous connection, BLE is only for short burst data transmission. BLE defines 40 usable channels as three primary advertisement channels and 37 data channels.

The biggest drawback of Bluetooth is the restriction to only one-to-one communication between two devices. That's why Bluetooth Smart Mesh was introduced in order to define and standardize a new mesh networking architecture for BLE. This improvement enables deployments of BLE for IoT. Bluetooth mesh topology works on a managed flood routing principle, with devices forwarding messages from one to another. The maximum number of devices connected in the Bluetooth mesh is 32767 [6, 2].

WiFi, standardized by IEEE 802.11 [29] and was commonly used for wireless local area networks. However, the first few generations weren't the best match for use in IoT. In 2017, the IEEE 802.11ah (WiFi HaLow) was introduced. A new version of Wi-Fi with extended coverage and low-power consumption requirements. Compared to other high-speed WiFi connections, IEEE 802.11ah aims to provide connectivity to thousands of devices and coverage up to one kilometer (which can be increased by using multiple overlapping hotspots), but with a maximum data rate of up to 347 Mbps.

ZigBee is a short-range wireless technology for wireless personal area networks (WPAN). It is a low-energy technology and is able to connect up to 255 devices with a maximum packet size of 128 bytes. ZigBee is able to cover up to 100m, but can vary to only a few meters based on the environment. Three types of devices are defined: coordinator, router, and end device. In order to make a big network, ZigBee can be extended as a generic mesh where local coordinators, to which devices are connected, are connected to global coordinators using multihop. The data rates supported in Zigbee are 20kbps- 250kbps [8].s

Radio-frequency identification (RFID) is a big part of IoT mainly because of its best usage, which is identification. RFID was first used in 1948 during World War II in Britain for identifying friendly equipment or personnel, enabling the distinction between allied and enemy forces. Since then, it has found usage in many other scopes such as Logistics and Supply, Manufacturing, Health Care and Medicine, Transport and Retailing, and much more [15].

RFID systems are divided into three main components: RFID tags, readers, and the application layer. There are two types of tags, active and passive. Active tags are battery-powered and communicate with other tags. Passive tags are powered by the readers using the coiled antenna, and the data is stored in the microchip. The reader activates the tags

and transfers the data to the application software. The application system processes the data and sends it to the host computer or stores it for later upload [18, 15].

Wired

Ethernet is a system connecting a number of computer systems in a local area network. Ethernet is commonly used for devices that don't need wireless communication, such as the sensors installed in a building automation. The high speeds ethernet can provide, using coaxial cables, twisted pairs or optical fibers, are good for a system that needs to send a large amount of data. Ethernet cables are good for transporting data over long distances, but the connection can be vulnerable to physical damage [21].

OBD is an Onboard diagnostic system used in the automotive industry for user-friendly access to information about the location and the actual fault in cars. OBD was first introduced by the "California Air Resources Board" (CARB) to control or reduce the air pollution caused by traffic, and also to give information about malfunctioning in vehicle systems by the Malfunction Indicator Lamp (MIL) being placed on the dashboard of cars.

If any malfunction happens in the car, the OBD gives a Diagnostic Trouble Code (DTC), which consists of five characters. These codes have some generic codes defined by the Society of Automotive Engineers (SAE), and manufacturers must also make a list of their own DTCs. These codes are accessed through an OBD kit connected to the laptop, where DTCs are shown in a user-friendly form [27].

Controlled Area Network (CAN) is a message based protocol [17]. CAN is a single or dual-wire bus, designed to connect and enable communication between electric components, microcontrollers, electronic control units (ECUs), sensors, devices, and actuators of the in-vehicle system [17]. It is a broadcast network that is able to get the speed up to one megabit per second [3].

When a CAN node intends to transmit data, it first monitors the bus to ensure it is idle. Upon confirming bus availability, the node initiates transmission by sending a data frame, which includes fields such as the start of frame, identifier, control, data, CRC, and acknowledgment. All nodes on the network receive the message, but only those for which the identifier is relevant process the data [3].

Collision Detection and Arbitration on Message Priority (CD+AMP) is used in CAN to resolve the collisions between two packets being sent simultaneously using message identifier bits [3]. The CAN frames, used for transferring the information between nodes [5], include an arbitration identifier (ID) field, used to determine the priority of the message; the higher the ID bit value, the lower the priority of the message [17].

2.3 Usage in Automotive Industry

Integrating the Internet of Things (IoT) into cars has transformed traditional vehicles into sophisticated, connected systems. With embedded sensors and communication devices, vehicles can collect and exchange data in real time, improving safety, optimizing performance, and enriching the user experience. For example, IoT enables predictive maintenance by monitoring vehicle health and alerting the driver of potential problems before they occur [26]. In addition, Vehicle-to-Everything (V2X) communication allows vehicles to communicate and interact with the infrastructure, which improves traffic flow and reduces the risk of accidents [39]. These advancements show IoT's vital role in the evolution of the automotive industry towards smarter and more efficient transportation.

The levels of automation used in cars range from advanced driver assistance systems (ADAS) to a fully automated driving system (ADS) [24]. Modern vehicles are equipped with various sensors that enhance driving safety and comfort. Different temperature sensors are used to monitor coolant, fuel, oil, and cabin temperature for the driver's comfort [10]. Pressure sensors are used for tire, fuel, and manifold pressure. Speed and acceleration sensors are used for ABS (anti-lock braking system), and accelerometers are used in stability controls and airbag deployment systems. Proximity and ultrasonic sensors are used for parking assistance [10].

Autonomous driving is now one of the newest inventions in the automotive industry. For ADAS systems to work, the car needs to sense its surroundings and make fast and intelligent decisions in real-time [4]. There are two types of sensors that autonomous vehicles need to use: exteroceptive sensors to see the outside environment and calculate the distance to it, and proprioceptive sensors to measure the values of the car [4]. Extroceptive sensors include: LIDAR (Light Detection and Ranging), used for measuring distances, radar, which can also measure distance, angle, and velocity of objects using radio waves, camera, to produce a digital image of the covered region, and ultrasonic sensors, also used for distance measurement [4]. Proprioceptive sensors are: GPS (Global Positioning System) to provide geolocation of the car, IMU (Inertial Measurement Unit) used for control and guidance of the vehicle, and Encoders to determine the relative position of the vehicle [4]. After collecting all the information from these sensors, the fusion of these data has to happen, as was stated before, there are multiple sensors that have the same function. Combining the data about the same obstacle but from different sensors can provide a higher quality output, so the system can perform at its best [4].

Vehicle networking is a big step to ensure road safety and efficiency. Vehicle-to-vehicle (V2V) and Vehicle-to-infrastructure (V2I) communications offer the ability to exchange speed, heading angle, position, and other information about the environment between vehicles and surrounding infrastructure [13]. V2V communication can help with many aspects of road safety. Sharing real-time data amongst the vehicles can provide information about traffic conditions to help drivers make informed decisions on route adjustments [38]. Accidents and hazard notification can also provide safety precautions, it can alert nearby vehicles about sudden braking, accidents, and road hazards [38]. V2V can also help with managing traffic flow by facilitating synchronized driving behaviors, such as speed adjustments and lane changes. When vehicles collaborate on the road, traffic flows more smoothly, reducing congestion and enhancing safety.

2.4 Motorsport and IoT

IoT has become a much bigger framework than when the first RFID-based applications were introduced. A wider scale of the sensors available, such as infrared sensors, laser scanners, and information sensors, together with a broader scope of connectivity, facilitating information flow and interaction between objects, has opened new possibilities in motorsports [35]. At the start of motor sports, data transmission was only available by wired connection, and later on, wireless communication was introduced to enable remote data transmission. The first wireless data transmissions started with radios, but later IoT technology has enabled the connection of sensors to cloud platforms via a wireless network for real-time data analysis and monitoring [35].

Data acquisition in motorsports can be used for many different reasons, but the main objective is to push the vehicle to the limits and get the full potential out of the vehicle and

to improve possible drawbacks that arise from the analysis [28]. The three main data categories that are being collected are: The vital functions of the vehicle, Driver activity, and chassis parameters. Vital functions of the vehicle can be engine oil and temperature, water temperature, fuel pressure, gearbox and differential temperature, and battery voltage [28]. This is the part of the data that can indicate any reliability issues with the vehicle. Driver activity parameters are those that the driver has control of; this data is later analyzed, and the driver uses it to learn from their mistake to improve their driving. Chassis parameters are mostly dynamics-related information, they can be parameters such as the vehicle speed, G-forces, steering angle, tire pressures and much more [28].

There are different approaches to analyzing the data. Depending on the type of racing, the data is being processed in different ways. For example, in Formula One, the data acquired in the car is being sent in real-time to the pit wall and also a remote data analysis center. The team analyzes this information instantly, allowing them to make strategic adjustments that enhance the car's performance during the race [31]. However, in the Rallye Dakar, the data from the vehicle is stored in the car for later analysis after the stage. But the main vital functions of the car are, of course, on the dashboard that the co-driver in the car has in front of him. There is valuable information about everything the crew needs to know to react in case anything unwanted happens. The visualisation is a big part of the operation because of the minimal time the co-driver has to react to what's on the dashboard.

Chapter 3

Data visualization

"Data visualization involves presenting data in a graphical or pictorial form, which makes the information easy to understand" [25]. Visualizations are made to help explain facts and take action based on the information visualized. It can be a powerful tool if the designer knows how to use it. Perception of the world around is the strongest sense humans have; approximately 70% of the sense receptors in human bodies are dedicated to vision [9]. Designers of the visual representation have to follow some rules to help this human sense with easier and faster perception. There are many types of data visualization, which is why one chunk of data can be visualized with many different styles, meaning that there are more approaches designers could use. There could be more different documentation on how to make the best data visualization.

Data visualisation can be applied in any field where presenting complex information is required to achieve different goals [25]. To make a data visualization that meets all the requirements, the designer must be guided by how the visualization will be applied. Here are a few applications mentioned in [25]. In public health, visualization is used to understand the problem better and take action where needed. In fraud detection, visualization is essential in the early stages of the investigation to detect fraudulent activity by studying patterns that suggest fraud. In the rally, data visualization is used to get real-time information about the car so the crew onboard can make informed decisions.

Based on the field in which the data is being visualized and the usage of the visualization, different properties of the visualization are enhanced. For example, dashboards, which are a visualization of data, are made for fast and easy comprehension of the data, and the data-ink ratio (which is explained closer in Subsection 3.1) has to be as high as possible [9]. However in a scientific paper where visualizations are used to explain a certain problem, the time of comprehension doesn't play as significant role as in dashboards that's why the information can be more detailed and the captions of visualization explain all the information the reader has to know to understand the visualization [19]. To achieve as good a visualization as needed, there are some principles that the designers should follow, and these principles will be explained in Subsection 3.1. Key Performance Indicators are used to better understand some collected data, as found in Subsection 3.2. To finally visualize and provide the data in a user-friendly way, Dashboards 3.3 are used. For the creation of the best possible visualization, there are some techniques and visualization media that will be shown in Subsection 3.4.

3.1 Principles of Effective Data Visualization

Data visualization can be a complex task. It is about transforming complex data into accessible insights through plots, clusters, or any other visual representation of data. There are many principles of effective data visualization based on the goal of the visualization. For example, in science articles, the data visualizations have to be as concrete as possible to provide complete information about the topic being presented. However, in dashboards, the principles are more focused on rapid data perception, showing as much information on a single screen, and consistency. Few [9] has created a complex manual of how to represent data in dashboards, what to keep in mind, and how to get the most out of the single screen. That's what is going to be discussed in this section.

Preattentive perception

Preattentive perception is one of the things that designers of Dashboards have to work with. It is a cognitive process of detecting and interpreting visual information instantly. If the visualization design is done right, human perception can automatically understand the information being presented without requiring focused attention. The preattentive attributes help with the data's efficiency and accuracy. Few have presented 11 preattentive attributes from Ware [36] book that are organized into four categories: color, form, spatial position, and motion.

The attributes of color are hue, which is the color itself; saturation, which is the intensity of the color; and lightness or brightness, which refers to how light or dark the color is. Intensity is a combination of both saturation and lightness, it determines the overall strength of the color. However, the color itself can be seen differently based on the surrounding colors, even if the value of the color is the same. That is why, in the design, the colors have to be contrasting to keep the readability as high as possible. One more thing that should be taken into consideration with color is color blindness. To ensure that even colorblind people will be able to read and see the differences between objects, the selected colors should be on a monochromatic palette so even colorblind people could see the difference. The differences between objects should be made with the same color but different saturations because the saturation of color is clearly distinguishable also by color blind people.

There is six attributes of form: orientation, line length, line width, size, shape, added marks, enclosure. All these attributes of the displayed data are used to draw attention, organize content, or encoding meaning. The orientation of text for example *italicized text* can make the italicized words stand out from the text. Line length and width are used in charts to describe quantitative data as bars in a bar graph, or if there is a underlined text, the width of the line below can draw more attention to the text with a wider line. The size of an object, chart, table, or icon can be used to declare greater importance to the part of the dashboard. Shape of objects can be used in graphs to differentiate datasets or in form of icons to assign distinct meaning. Added marks can appear next to important information in the dashboard. And lastly enclosure to group section of data, or highlighting content as important.

The attribute of position is a 2D position to encode quantitative data in a graph, and it is the easiest and most accurate, preattentive attribute to perceive. And the attribute of motion, flicker. Flickering objects on the screen draw attention to themselves. Even though the flickering objects can be annoying, they might be helpful sometimes.

Few [9] has also stated that two different types of colors should be used for the dashboard design, standard colors and emphasis colors. Standard color pallet should be used to provide calming, neutral backdrop for the dashboard. Standard colors are soft and natural inspired by natural tones. Its colors such as soft grays, browns, oranges, greens, and blues. On the other hand the emphasis colors are the one that are highly saturated, bring attention to them selves. These should be used to emphasize something on the board, something that needs to be standing out.

Gestalt principles

Other principles that are used to make visual perception as effective as possible are the gestalt principles. Gestalt is a german word which means pattern in English. The researchers from the Gestalt School of Psychology has recognized that our brains organize stuff that we see to understand it. Gestalt principles of perception reveals visual characteristics that incline us to group objects together. The information about this principles are from the Few book [9] and [32].

The **principle of proximity** is based on grouping objects just by putting them close together. As you can see in Figure 3.1 circles that are closer together (the distance between them is approximately the same) are perceived as groups. That is why this is the easiest way of grouping data on the dashboard.

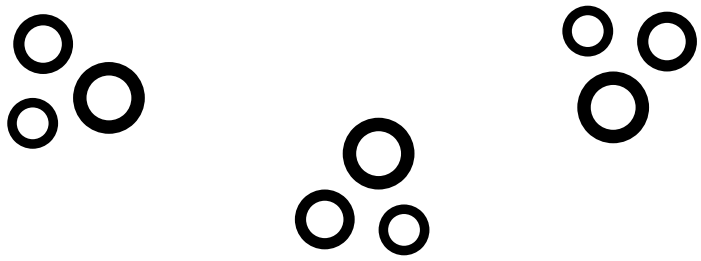


Figure 3.1: Grouping the circles using proximity principle. Adapted from [9].

Other principle used for grouping objects together is called the **principle of similarity**. Principle of similarity is based on the fact that we tend to group objects similar in color, shape, size, and orientation, as is shown in Figure 3.2. This principle can help group related data on the dashboard that are in different places. This principle is similar to using color to group things together, but that is usually used to connect the same information in different graphs. However, similarity can be used to compare data that are in different places.

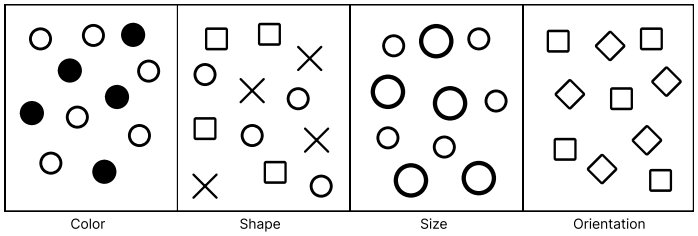
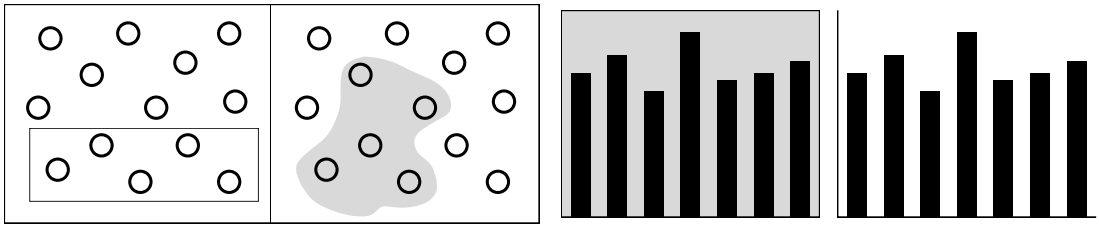


Figure 3.2: Creating group based on similarity. Adapted from [9].

The **principle of enclosure** is simply a way we perceive things are grouped together if they are enclosed by anything that creates a border around the objects. It could be either a line or a different color base as shown in Figure 3.3a.

The **principle of closure** is a way to group objects with incomplete, open, or any other unusual form. Humans perceive these objects as closed or complete if there is a reason to do so. This principle can be applied to dashboards to make graphs, as is shown in Figure 3.3b. This could be a way of avoiding unnecessary visual content to keep the dashboard as clean as possible.



(a) Enclosure (b) Closure

Figure 3.3: Examples of using enclosure and closure for grouping objects. Adapted from [9].

Another one of Gestalt principles is the **principle of continuity**. We perceive objects that are aligned together as a group. That's why a dashed line is perceived as a line rather than many short lines. This principle can be useful in Dashboards when trying to align some objects together, for example, in a table as is shown in Figure 3.4.

Division/Department	Headcount
G&A	
Finance	15
Purchasing	5
Information Systems	17
Sales	
Field Sales	47
Sales Operations	10
Engineering	
Product Development	22
Product Marketing	5

Figure 3.4: Even when the columns in this table are not enclosed with lines, the alignment clearly groups the columns. Redrawn from [9].

The final Gestalt principle is the **principle of connection**. This principle uses a way of making groups by simply connecting them with a line. An exact connection can make a stronger connection between two objects than the ones produced by proximity or similarity. This technique connects non-quantitative data, such as connections between employees.

3.2 Key Performance Indicators (KPIs)

Key Performance Indicators (KPIs) are the critical (key) indicators of progress towards an intended result. KPIs are measurable values that indicate how effectively the goals are being achieved. They play a significant role in monitoring, assessing, and optimizing performance. Here are some examples of KPIs to better explain how they can be implemented in different fields. In finance, managers can choose from many different indicators to track their financial progress, such as gross profit margin, net profit margin, and working capital ratio. In IT, KPIs such as Total Support tickets, IT costs vs revenue, and Security-related downtime can be used. In motorsport, the performance of the car is monitored with these indicators: Lap times, speed and acceleration, fuel efficiency, and much more, which will be discussed in 4.1. So as the examples show, it is crucial to set the best KPIs that reflect the progress and overall effectiveness of the activities done.

Mikušová [20] provides multiple divisions of indicators that will be discussed in this paragraph. The indicators can be divided into groups based on different aspects. However, the three divisions that will be further discussed are: reproducibility of the use, subject of measurement, and area of measurement.

The indicators are divided into two groups: continuous and discrete, based on their reproducibility. Continuous indicators are those types of indicators that exist on the infinitely divisible scale and or continuum, which means that the units in which the indicator is measured can be divided into smaller and smaller units, for example, time can be measured in hours, minutes, seconds, etc. These indicators are being measured repeatedly at predetermined times. The discrete indicators on the other hand represents characteristics with distinct, separate values, such as description, frequencies, or categorical scales, for example, descriptive categories could be levels of education (primary, secondary, bachelor's), frequencies is for example number of processed orders, and evaluative scales could be ratings like "excellent," "satisfactory," "unsatisfactory." These indicators are being observed repeatedly at set intervals for intangible quantities, such as customer satisfaction.

The subject of Measurement divides indicators into hard and soft indicators. Hard measures are measurable, objective metrics that can define a company's performance or activities in areas that directly impact competitiveness. Hard indicators are quantifiable, which means that the indicators could be measured precisely and objectively, for example, revenue, profit, and costs. They should be easily available and shouldn't create additional costs. Finally, they should be able to be expressed in financial terms to be easy to compare. The purpose of hard indicators is to define specific targets. Soft indicators are the complete opposite of hard indicators. They are subjective measures to assess the qualitative aspects of performance. They are difficult to quantify and non-monetary, which means that they can't be expressed in financial terms. They are often based on judgments, surveys, or audits. The purpose of soft indicators is to evaluate areas like satisfaction, quality, or other aspects.

There are four different groups of indicators, which are divided based on the area they measure. These areas are: efficiency, effectiveness, result, and process. Efficiency indicators are used to evaluate how economically resources are utilized or find ways to reduce costs while maintaining output quality. Indicators of effectiveness measure the level to which customer needs are met. Indicators of the result describe the outcome of the process. Finally, indicators of process are used to monitor and improve internal processes that lead to the result. Here, the importance of measuring not only the process but also the outcome can be seen. By integrating efficiency, effectiveness, result, and process indicators into

a performance measurement system, organizations make improvements and maintain the quality of customer service.

3.3 Dashboards

As defined by Stephen Few [9], a dashboard is a "visual display of the most important information needed to achieve one or more objectives, consolidated and arranged on a single screen so the information can be monitored at a glance." However, this is not the only definition, there is a broader definition from Wexler et al. [37] "a visual display of data used to monitor conditions and/or facilitate understanding." Combining these two, the meanings and purpose of dashboards can be explained as a single screen that provides needed information about a condition related to a specific task [14]. Dashboards are mainly used as a tool for gathering data from various sources, enabling monitoring, analysis, and real-time decision-making.

Stephen Few [9], identifies several key characteristics that make dashboards effective tools for data visualization and decision-making. The main reason is that the dashboard has to be a single-screen data visualization that allows users to see everything at a glance. The information on the screen should have real-time or near real-time updates to keep the timely decisions as accurate as possible. The screen should only display the most important information, such as KPIs, to keep the screen free of unnecessary data that only slows the perception time. The effective dashboard should focus on simplicity, avoiding unnecessary decorations and using size, color, and position to highlight important information. Above all, dashboards should be made to help users quickly assess situations and make informed decisions.

Dashboards are versatile tools used across various industries to monitor and analyze key data in real time. In business, they track KPIs such as sales, revenue, and employee performance, which facilitates informed decision-making. In healthcare, dashboards assist in patient monitoring, hospital management, and tracking medical supply inventory, enhancing operational efficiency and patient care. In transportation, they are utilized to track fleets, monitor traffic, and optimize routes, improving logistics and reducing transportation time. In the automotive industry, dashboards display real-time vehicle data, such as speed, fuel status, and diagnostics, which aids in vehicle maintenance and performance evaluation. These applications demonstrate the adaptability and significance of dashboards in transforming data into actionable insights across various sectors.

3.4 Visualization Techniques in Dashboards

To create a dashboard that will meet our requirements, we need to carefully pick the best display media that will enable clear and immediate communication. When trying to choose the best display media, the nature of the information and message must be considered. To present the information effectively, the layout of the dashboard has to be organized in a certain way, which will be discussed later. In this section, these problems will be addressed. The information in this section is from the Few [9] book.

Dashboard Layout

When designing a dashboard, the information layout plays a significant role in the fast perception of the data. It's another way of emphasizing the information. Two types of data

are shown on the dashboard: Information that is always important and information that is only important at the moment. The designer should divide the information that is going to be on the dashboard to know where to place it. For this placement, Few has divided that dashboard into five different areas, as u can see in Figure 3.5. The data that are always important should be placed in the top left corner or in the middle, however the information in the middle is only emphasized when distinguished from the other data by a border such as a line or a white space.

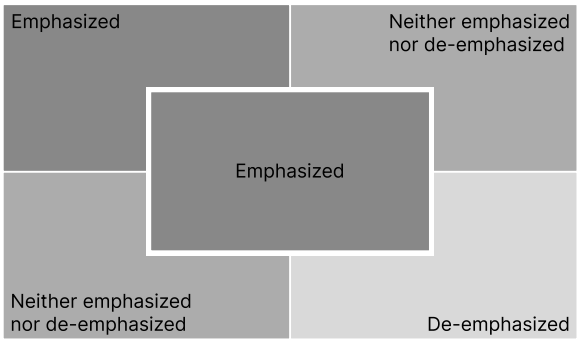


Figure 3.5: Different areas on the dashboard have different levels of visual emphasis. Redrawn from [9].

Display media

To display our information in the most effective way, the proper display media has to be picked. Many different display media exist because every type of information should be presented in a way that best fits the message. Two principles proposed by Few guide the selection of the media: It must be the best means to display a particular type of information commonly found on dashboards, and it must be able to serve its purpose even when sized to fit into a small space. Few have also proposed six different types of media: graphs, images, icons, drawing objects, text, and organizers.

Graphs

Graphs are the most commonly used display media in dashboards. This is mainly because of the quantitative data usually displayed on dashboards. Few [9] explains nine types of graphs which are going to be discussed in the next part of this section: bullet graphs, bar graphs (horizontal and vertical), stacked bar graphs (horizontal and vertical), line graphs, sparklines, box plots, and scatter plots.

Bullet graph is Few's invention, made mainly for dashboard usage; you can see the design in Figure 3.6. The ratio of information and space occupied by the bullet graph is what makes it really suitable for the dashboards. It is designed to display a key measure, with a comparative measure and qualitative ranges, to be able to instantly declare if the measure is good, bad, or in some other state. The intensity of the color distinguishes the qualitative range to ensure that even a color-blind person can see the differences. The qualitative range should be divided into more than five states to clarify the colors' distinctions. There is also a way to display more than one comparative measure simply by adding another distinct marker to the graph. To ensure distinctions, the markers can have different line widths.

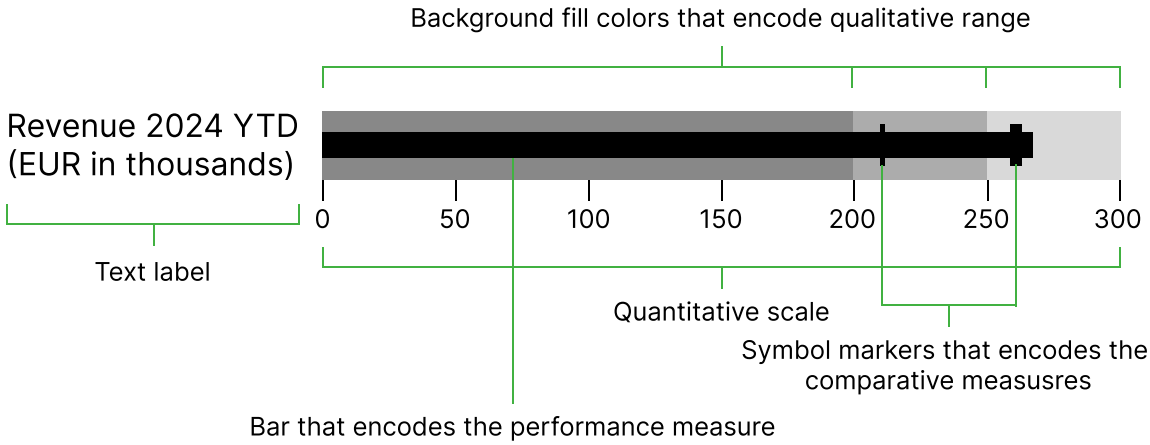
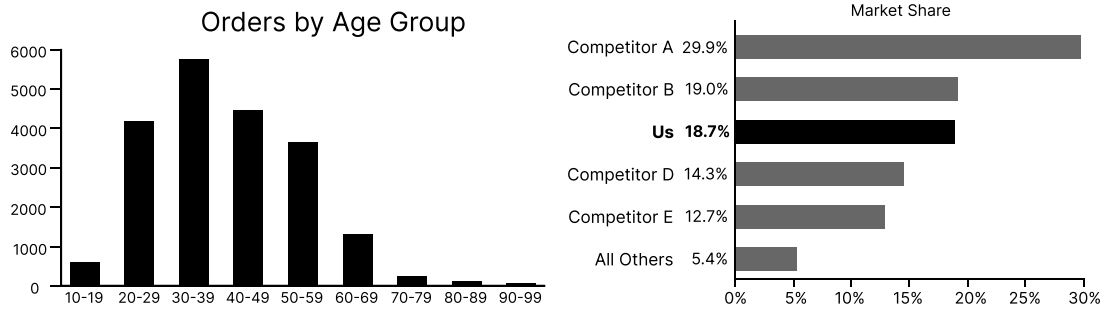


Figure 3.6: Graphic explanation of bullet graph. Redrawn from [9].

Bar graphs are used to display discrete data categories. The bars in the graph represent the exact quantitative values, which means that they can be easily compared to the other values around by simply comparing the height of the bar. They could have two ways of orientation, which are vertical 3.7a and horizontal 3.7b. The bar graphs are used to display data of nominal and ordinal scales. The nominal scales are composed of discrete items in the same categories but don't have any relation, for example, different departments (Sales, Marketing, and Finance), regions (The Americas, Asia, and Europe). The ordinal scales consist of items that have a relation between them but don't declare any quantitative information on their own, for example "A, B, and C" or "small, medium, large".

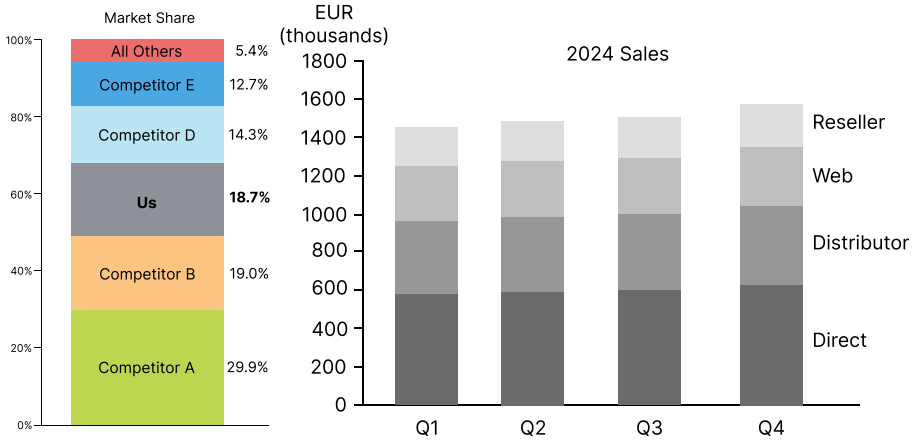


(a) Vertical

(b) Horizontal

Figure 3.7: This figure shows the two possible orientations of a bar graph. Adapted from [9].

Stacked bar graphs is a variation of the bar graph. The usage of this graph is misused in most cases. Stacked bar graphs tend to be used to display a single series of part-to-whole data. However, that is not the best use case. Because when you look at the stacked bar graph in Figure 3.8a, it takes much more time to read the information of the stacked bar graph than from the horizontal bar graph that can be seen in Figure 3.7b.



(a) Stacked Bar

(b) Stacked bar graph

Figure 3.8: Examples of using enclosure and closure for grouping objects. Adapted from [9].

The only use case that Few says is suitable for stack bar graphs is when the graph must display more instances of the whole, and the graph is more emphasized on the whole than the parts of the whole. An example can be seen in Figure 3.8b.

Line graph is best for displaying the shape of data, its movement between values, and its overall change over time. Line graphs can be very useful for showing trends, fluctuations, cycles, rates of change, and how the datasets vary in relation to one another. Usually, when showing time-series data on a dashboard, the shape of the data is what matters, not the detailed values that can be shown in a bar graph. In Figure 3.9, the different information that can be read from the graphs that show the same data can be seen. The bar graph on the left is more emphasized on the individual values, and the line graph on the right side clearly shows the overall shape of the data. Line graphs don't need to start from zero, as they show only the highest values of the data; that's why the graphs can be more detailed.

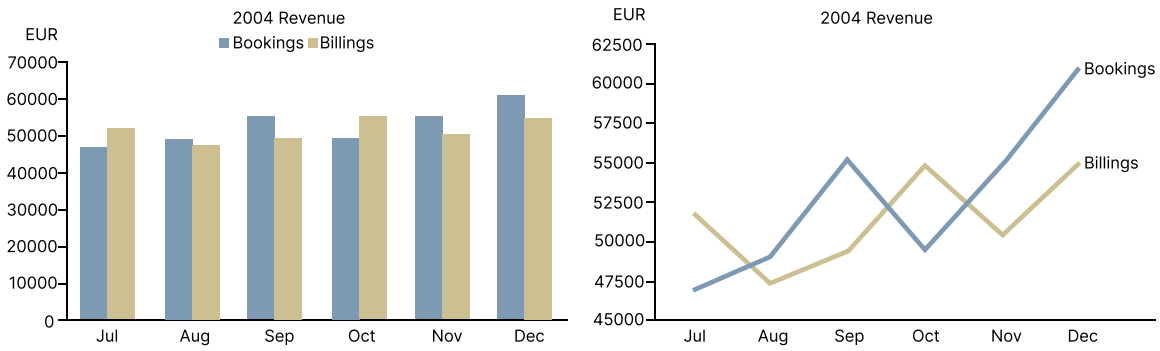


Figure 3.9: The information that is displayed on the bar graph is different from the line graph even when showing the same data. Redrawn from [9].

Sparkline graphs are used to display the historical meaning of the value. That means that there is no quantitative scale provided with these graphs, that's because they are not used to show any detailed data. The main purpose is to show how the values have been changing over time. The sparkline in Figure 3.10 on the left is showing a 12 months history of a checking balance, as you can see it is clear to see how the values have been changing,

and if it has gone down or up in the shown time. However it is possible to show a bit more information. As is seen in the sprakline on the right side of Figure 3.10, there is a gray rectangle behind the sprakline, which indicates the number of manufacturing defects that are acceptable. The red dot at the end of the sprakline ties the current value, which is represented by the red five on the side.

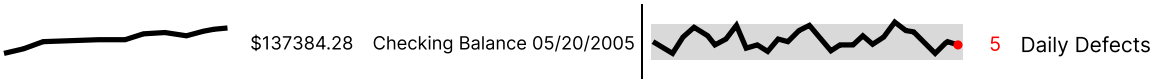


Figure 3.10: Sparklines example. Redrawn from [9].

Box plots display a dataset's distribution across its entire range. They help visualize the data distribution and identify the median, quartiles, and potential outliers. So, for example, when trying to display the distribution of salaries across multiple salary groups, it can be like in the graph on the right side of Figure 3.11. On the left side, the box-and-whisker plot itself is explained. As is seen, the box-and-whisker plot consists of a line representing the distribution's median, whiskers representing the highest and lowest values, and the box representing the main 50% of the distribution. The median divides the values in the distribution into two halves, which means that if the median line is closer to the top, more values are in the upper half of the distribution, and if the median is closer to the lower end, it's the other way.

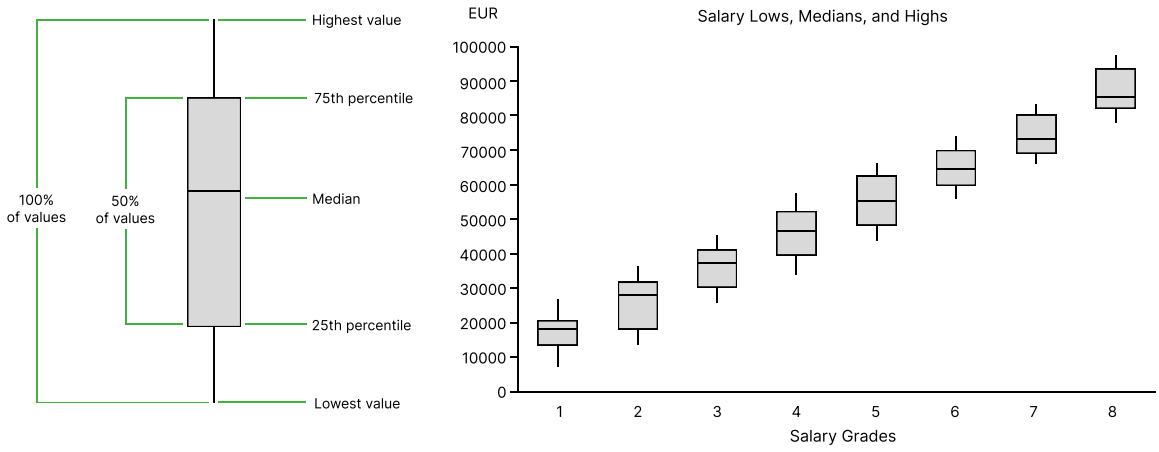


Figure 3.11: Graphic explanation of box plot and example of usage. Redrawn from [9].

Scatter plots have only one usage, which is displaying the corelation between two datasets. It can display the direction in which its correlating, and to what degree the two paired data sets are correlating. Figure 3.12 is showing a correlation between the Ads and the sales revenue. This data has been collected for 24 months and each dot in the graph represents one month. The line represents that the correlation is positive (going upward from left to right), which means that the bigger the number of ads the bigger the sales revenue. Judging by the grouping of the data values around the trend line, the correlation is strong, so it can be said that the number of ads strongly affects the sales revenue.

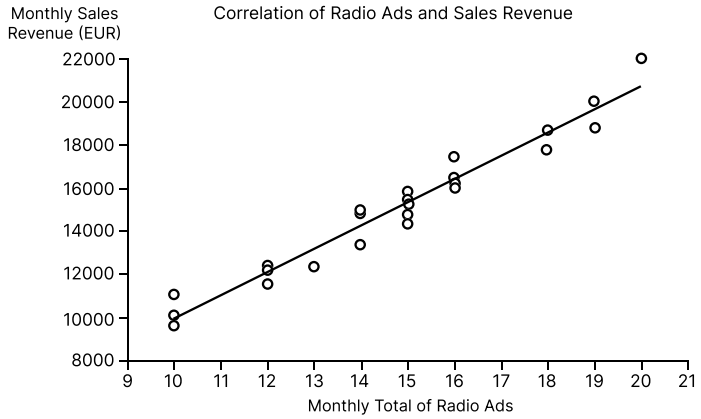


Figure 3.12: Example of scatter plot. Redrawn from [9].

Icons

Icons are mainly used in dashboards to draw attention to a piece of information. This is best used when alerting to something wrong that requires attention. Alert icons should not be visible all the time, they should only appear when its needed. So, for example, it is not that effective when the icon is always displayed only the color changes from green to red. When the color is green, that means everything is going fine, so there is no point in displaying the icon. The alert icons work best when they are displayed only when the problem arises, that is because when something is added to the dashboard that wasn't there before, it's preattentively perceived as an "added mark" and that doesn't work the same when the color only changes from green to red.

There are two more types of icons that Few has shown in his book, **up/down icons** and **on/off icons**. The up/down icons are used to demonstrate that a value is going up or down compared to some other value or value in the past. The on/off icons distinguish some items on the dashboard from others, for example, marking featured items in a list. Many types of marking icons exist, but the most used are asterisks, checkmarks, or Xs, as shown on the right side of Figure 3.13.

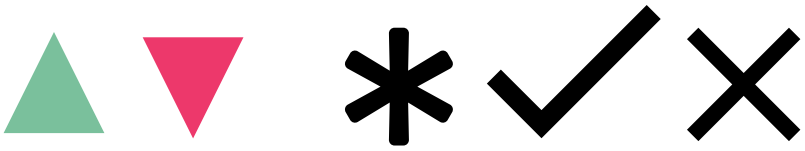


Figure 3.13: Up/down icons on the left, On/off icons on the right. Redrawn from [9].

Text

All of the dashboards always needs some text. Even though they should be graphically oriented, some information must be displayed as text. Not even the graphs and other types of graphical visualization can work without some textual description. Text representation is best when it doesn't need to be compared to something else; it's standalone information.

Drawing objects

Drawing objects can be used to clearly show relationships between objects on the dashboards. For example, when displaying a process, arranging events in a sequence, and using arrows to connect the steps to indicate the flow of the process. It's much easier to understand with the connections. It can also represent the connection between entities by using circles or rectangles for the entities and arrows or lines to connect them.

Chapter 4

Dashboards in Racing Specials

The dashboard in racing specials is one of the car's most essential parts. It's the only way the passenger or driver can communicate with the car. There are numerous types of dashboards. Some of them are meant for the driver, and some for other passengers, for example, navigators or mechanics. To ensure the best possible performance of the car, a lot of KPIs have to be monitored and also visualized on this dashboard. In this chapter, all of these problems will be discussed, along with an analysis of some existing solutions. There will be three commercially available dashboards for general purchase and one custom-made for the Dakar truck.

4.1 Key Performance Indicators used in Racing

The vehicle must be in its best shape to perform well in the race. The vehicle's status is usually monitored by a large number of sensors all around the vehicle. These sensors provide numbers and statistics that could be considered the KPIs of the racing vehicle. There are many different indicators in racing vehicles. Some are mainly for the driver, such as the speed, RPMs, selected gear, etc. But some data have to be examined by other team members, like mechanics on the pitwall (in the circuit racing) or mechanic in the car (Rallye Dakar truck racing), the data are, for example, the temperatures of different parts of the vehicle, pressures of liquids, and much more. To ensure the best performance and reliability of the car, all of these KPIs should be monitored and analyzed during the race to prevent any malfunction or, in the worst case, detect a malfunction.

Since this work is focused on the dashboard that the mechanic in a rally car has, we are going to take a closer look at the KPIs that the mechanic needs to monitor. The information about the vehicle can be divided into six groups: Fluid pressures, temperatures, wheel, tire, and suspension monitoring, electrical system, engine diagnostics, alert, and error system.

Numerous fluids in the car ensure the car's functionality, including engine oil, fuel, brake fluid, and the hydraulic system. The pressure of these fluids significantly impacts the vehicle's performance. Engine oil ensures that the engine is properly lubricated, and if the pressure drops, it could mean that there is a leak in the system. Detecting this could prevent severe damage to the engine. Fuel pressure ensures that the fuel is injected into the valves in the required quantities. Brake fluid pressure confirms the braking system's responsiveness and helps detect a leak in the system. The hydraulic system is all about pressure and helps with power steering or suspension.

Temperatures of different system parts have to be monitored all the time. Engine coolant temperature has to be monitored to prevent overheating and severe damage to the engine. Transmission temperature is highly correlated with the transmission's performance and longevity. Differential temperature monitoring can help the mechanic prevent issues and also ensure functionality. If the oil in the differential gets too hot, it could break down, leading to damage. Brake temperature is helpful to avoid brake fade (malfunction of the brakes), which happens when the temperatures are too high.

Wheel, tire, and suspension monitoring is done to determine if the tire maintains the pressure it is supposed to have. Wheel angle helps with better alignment and also could help with early detection of damage to some crucial parts of the wheel hub, like axles or knuckles. Suspension travel monitors the impact of the terrain on the suspension, which can help with better tuning of the suspension. Damper temperatures are needed to make sure that the temperature is in the right range in which it is functioning the best.

In the electrical system, the battery voltage and current have to be monitored to know if the system is working as it should. Also, the alternator performance is much needed to know if the battery is effectively charged.

The engine monitoring focuses on the air-to-fuel ratio to ensure optimal combustion efficiency, and the turbocharger is also monitored to determine the boost level and temperature to prevent and detect any damage or malfunction.

Lastly, the Alert and error system helps identify and address system malfunctions in the form of DTCs discussed in the 2.2 section. Warning indicators alert the mechanic to any malfunction of the whole system.

4.2 Purpose of the Dashboard

As the previous section outlined, the dashboard, which is the main product of this work, is mainly for the mechanic who is onboard with the driver and navigator. This dashboard is supposed to provide information about all of the vital parts of the car so that the mechanic can ensure the driver that the vehicle is running correctly or to analyze what could be the source of the current problem the driver is experiencing. Since the Rallye Dakar is a high-speed race in dunes, which is really harsh terrain, it has to be as simple as possible so that the mechanic can easily withdraw data from the dashboard. The display has to provide all the information needed in a fraction of time and/or in really bumpy conditions.

The whole dashboard is mainly used to resolve any unwanted situation. This could be some malfunctions of any part of the vehicle, loss of power, overheating, or anything else. These problems need to be discovered as soon as possible in order to prevent any terminal damage, which could lead to DNF (did not finish) of the race.

4.3 Requirements for the display

To pick the most suitable display for our cause, we need to define some requirements that will help us with this decision. There will be four main things that will be considered when picking the proper hardware. Environmental durability, Screen size and resolution, the whole dimensions, buttons available for navigation between screens, and customization of the screen layouts and alerts to provide the best possible design of the data visualizations.

Since the conditions at the Dakar rally are so harsh, the display has to be durable to withstand a lot of dust, heavy heat, and impacts from the terrain. To provide good visibility

during the race, the display has to provide good brightness and ideally an ambient light sensor to adapt to the ambient light.

Size and resolution are significant deciding factors. The dashboard will be displaying a lot of data, which is why it will be ideal if the display has 12″ or the other possibility is to use two 6″ displays. The second option, however, has more downsides since the display made for these conditions usually has some cover which can take up a lot of space. That's why the 12″ display is a better option. To provide the best usability for the mechanic, some buttons for navigation between the different screens and manuals should be available; if not, the display should have some possibility to connect external buttons.

The most essential requirement for the display is the customization of the layouts and alerts. Since this work is all about fast perception and good readability, the display has to provide good options for custom layouts. Usually, these displays have some software where the display could be customized, so this must be considered.

4.4 Analysis of existing solutions

This section will analyze and compare three different displays from different companies. First, each display will be presented separately, and then the comparison will be made. The comparison will be made based on the requirements in Section 4.3. All of the information in this section is from the official pages of the companies that made these displays. All of these analyzed solutions are commercially available solutions.

MoTeC C1212

Figure 4.1: MoTeC C1212 display and the MoTeC display creator used for layout customization [22]

In Figure 4.1, a full color 12″ display from the MoTeC company can be seen. The manufacturer states the readability in direct sunlight is outstanding, and the display is described as "ultrabright" and "anti-reflective". The exact luminance value wasn't mentioned in any documentation. The resolution of 1280x480 pixels provides a wide display where the information can be clearly shown with a lot of space to work with.

Multiple connection possibilities are available on this display: CAN bus, RS232, and Ethernet. This can provide an easy connection with ECUs or other vehicle systems from which the information should be retrieved. There is also a possibility of multiple expansions for this display, one of them being a *data logger*, which could increase the number of monitored parameters.

The IP65 rating that this display has is good evidence of the capability to withstand the harsh conditions of the Dakar rally. It is built to be protected from the environment that rally racing provides, like extreme temperatures, dust, lasting vibrations, and hard impacts from the uneven terrain.

Customization on this display is made with an app provided by the manufacturer, which is called MoTeC Display Creator, which can be seen in Figure 4.1. The app provides a simple UI in which all the inputs from CAN and other connections can be prepared, and also the layout of the display can be configured with multiple types of display media. However, you cannot design your own media, which can limit the possibilities of the visualization. The app also provides a possibility to simulate the dashboard, which could be helpful for testing and customization to meet the user's requirements. There are also some prepared layouts, but because of the specific use in this work, they are not suitable. All the layouts that are available in the app are not appropriate for the cause of this work since they are all made for the driver directly. Every layout has information mainly focused on the driver, which means that not all of the vehicle's information is available. That's why all the layouts would have to be made custom with a better focus on the data that was mentioned in Section 4.1

ECUMASTER ADU5/ADU7

Figure 4.2: ECUMASTER ADU5/ADU7 display and the ECUMaster Client software used for layout customization. Retrieved from [7].

This display from Ecumaster, which can be seen in Figure 4.2 comes in 2 different sizes, 5″ and 7″. With the resolution of 800x600, it could be suitable for the second variant that was proposed before. The High-brightness and anti-glare coating on top of the display ensures good visibility even in direct sunlight. TFT LCD display ensures the colors are vibrant and sharp, which is highly needed in the Dakar rally.

Ecumaster ADU provides a CAN connection along with eight analog and four digital inputs, which could be used to connect external buttons since this display doesn't have any. The ECUMaster manufacturer also provides a CAN-based keyboard that could be used with this display. The display is compatible with a wide range of ECUs and CAN-based devices, which could help integrate this display into the car's already existing structure. ECUMaster also provides modules to enable onboard data logging.

High durability is backed by the IP65 rating, similar to the MoTeC 1212 display. Since this display is also made for racing, it can withstand the conditions without problems.

Customization on this display is done in the app provided by the manufacturer called ECUMaster Client software, which can be seen in Figure 4.2. Multiple screens are provided in the app that can be further customized, or the user can make a custom one. The display

supports up to 10 different pages that could be switched between automatically or with the external keyboard connected. The display media that can be used with this display look advanced, but the selection isn't as big as with the first display presented. The one thing that this software stands out in is the statistics that the logger can provide. The driving style and much more information could be easily analyzed using this software's graphs.

AIM MXT Strada

Figure 4.3: AIM MXT Strada display and the RaceStudio3 software used for layout customization. Retrieved from [1].

In Figure 4.3, a 10″ display from company AiM Technologies can be seen. It is one of the MXT series of displays from AiM, which means it also comes in a smaller version, but there could be some limitations. The display provides a resolution of 1280x480 pixels and brightness of 800 $\mathrm{cd/m^2}$ can ensure excellent visibility in direct sunlight. The contrast this display provides is 1100:1, which provides nice, vibrant colours for data visualizations.

The MXT provides multiple connection possibilities. Eight analog and four digital inputs are available for data inputs. For communication with ECUs CAN, RS232, or K-Line connections can be used, which can ensure compatibility with a broad selection of ECUs. The MXT also provides capabilities for additional modules, which can offer more possibilities for data acquisition and functionality. The display has four side buttons that enable navigation between the different screens.

The durability rating on this display is slightly higher than on the previous ones, it comes with IP67 rating. That means the display can withstand the conditions of the Dakar rally. Also, the aluminium body provides protection against any damage.

The customization on this display is made in the app called RaceStudio3, which can be seen in Figure 4.3. This app provides customization of all the channels coming to the display, icons on the display, the shift button can be pre-programmed to be used for predictive time, and much more. However, there is a big downside when it comes to customizing the layout on the screen itself. This display does not support custom layouts. The user can only choose from different presets that are available and customize the data and numbers that are shown. Six RGB LEDs can be customized to have various functions. To enable data logging, this display comes with 4GB internal storage, which could come in handy in the Dakar rally.

Comparison

To provide a clear comparison between this three analyzed displays the table 4.1 provides all the technical information of the displays.

Feature	MoTeC C1212	AiM MXT	ECUMaster ADU
Display Size	12″	10″	5″ or 7″
Resolution	1280x480	1280x480	800x480
Brightness (cd/m²)	Unknown	800	High-Brightness
Contrast Ratio	Unknown	1100:1	Unknown
Day/Night Mode	Yes	Yes	Yes
Analog Inputs	N/A	8	8
Digital Inputs	N/A	4	4
CAN Channels	2	2	2
Other Connections	RS232, USB, Ethernet	RS232, USB	RS232, USB
Customization Software	Display Creator	Race Studio 3	ECUMaster Client
Custom Layouts	Yes	Limited	Yes
Predefined Layouts	Yes	Yes	Yes
Custom Graphics	Yes	Limited	Yes
IP Rating	IP65	IP67	IP65
Material	Aluminum	Anodized Aluminum	Aluminum
LEDs for Alerts	No (on display)	10 RGB LEDs	RGB LEDs
External button support	Yes	Yes	Yes
Multi-Page Support	Yes (Customizable)	Limited	Yes
Data Logging	Optional (Add-on)	Yes (4 GB)	Optional (Add-on)

Table 4.1: Comparison of Displays for Dakar Rally Truck

Custom made solution

Figure 4.4: Custom made dashboard used in Rallye Dakar truck.

This is a custom-made dashboard that has already been used in a Rallye Dakar truck. This dashboard is made on a display with all the requirements to withstand the harsh conditions

of the Dakar Rally. It has a high brightness, which can provide good visibility in direct sunlight. The display also has 12 buttons, each mapped to a different function, and one dial that is used to edit the speed limit of the car. The display is connected to a computer that is processing all the information from the vehicle's sensors, which means that the data comes to the display in an already user-friendly form.

The design of the data visualization is made with simple gauges, where each value has its gauge with the same design. In the left part of the dashboard, a simple visualization of the vehicle is displayed with all six wheels available on the car, including two reserve wheels. The visualisation of the data on this dashboard is very simple, and the use of the same visualization media for each of the values could confuse the user. The gauges don't provide any reference or anything else that could help the mechanic with a quick check if the values are in the ideal range. This is a significant downside of the dashboard. It is not really built for fast perception, and the visualization media is outdated.

Chapter 5

Proposal

In this chapter, a proposal for a modern fast-perceptive dashboard will be made. The proposal is made for a dashboard used by mechanics in a rally car. The main goal of this proposal is to make data visualization in a style that enables fast perception and doesn't include any unnecessary information or graphics. It's designed to provide as much information as possible in a short amount of time. The proposal is made for a 12″ display and 1280x480 pixels.

The purpose of this proposal is to replace the old-fashioned visualization styles. The dashboards available now on the market, as can be seen in Section 4.4, provide a good looking dashboards with information prepared for the driver, using gauges and other styles of visualization that are not as fast perceptive or doesn't provide the level of detail needed. Most of the dashboards from different companies are more focused on the style than the usability. They include a lot of non-informative ink, which can prolong the time of perceiving the information and takes much more space than needed. The replacement data visualization display will be explained in the next sections.

This proposal will include two different dashboard designs. First is a design made only with the information this work provides. The display media used were selected based on the assumptions of what could be the best for the mechanic and what could provide the ideal amount of information and detail. The second design that will be proposed is the edited dashboard design after consultations with Ing. David Svída, Ph.D., and Ing. Martin Beran, engineers from one of the Rallye Dakar teams, who have a lot of experience from rally racing.

5.1 First design

Since the dashboard is made for the mechanic, it visualizes only information about the live condition of the car. It is divided into seven parts where each part is designated for different types of information, for example as can be seen in Figure 5.1, in the top left part of the dashboard the temperatures of the different parts of the car are visualized and in the top right corner the pressures are. This way, the dashboard provides information of the same context in the same place, so the user doesn't have to look all over the display. The most needed information on this dashboard is represented by visualizations, not only numbers, so a quick look at the dashboard can provide the needed information.

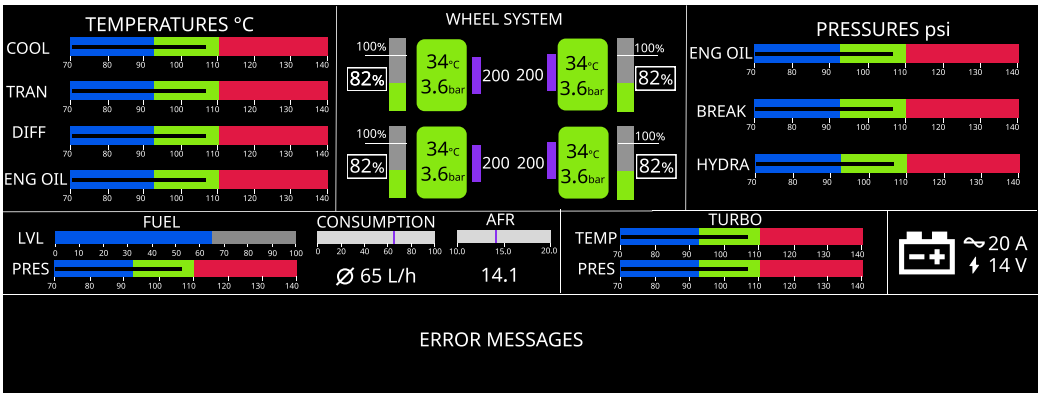


Figure 5.1: Proposed dashboard

Temperatures

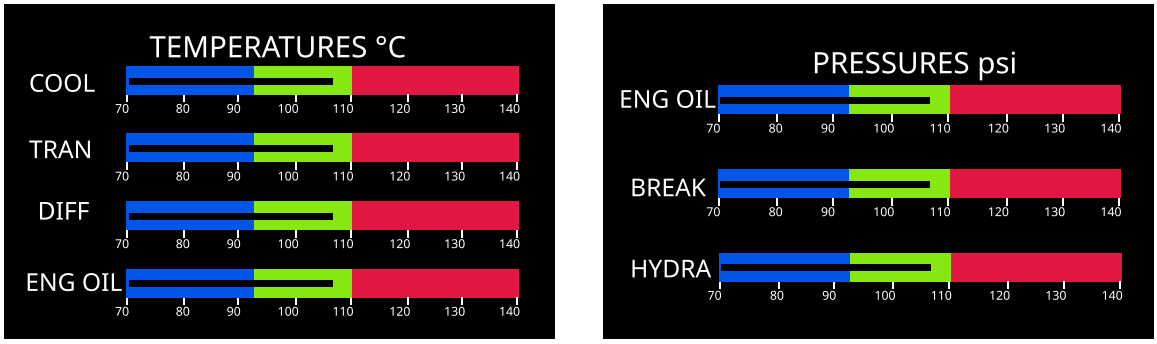


Figure 5.2: Part of the dashboard where the temperatures are displayed.

In these two sections, the temperatures and pressures of different car parts are visualized. I used the bullet graph, which is explained in Section 3.4. The display media used for temperatures in most of the existing dashboards are usually a classic bar graph, or in the worst case scenario, the thermometer graph, and the pressures are usually visualized by gauges. The bar graph could work if the user doesn't need as much detail, but in a situation such as a rally car, the detail is much needed. The user needs to know the comparison to the range in which the temperature needs to be, which the bar graph doesn't provide, but the bullet graph does. The problem with the thermometer graph is that it also doesn't provide any reference and takes up a lot of space. Gauges for the pressures can be a good visualization medium, but they take up a lot of space, and that's why the bullet graph is also better for this. The shortcuts next to the graphs, which represent the information the bar graph is visualizing, are also color-coded; the shortcuts change colors according to the value, which can help with faster discovery of a problem.

Wheel system

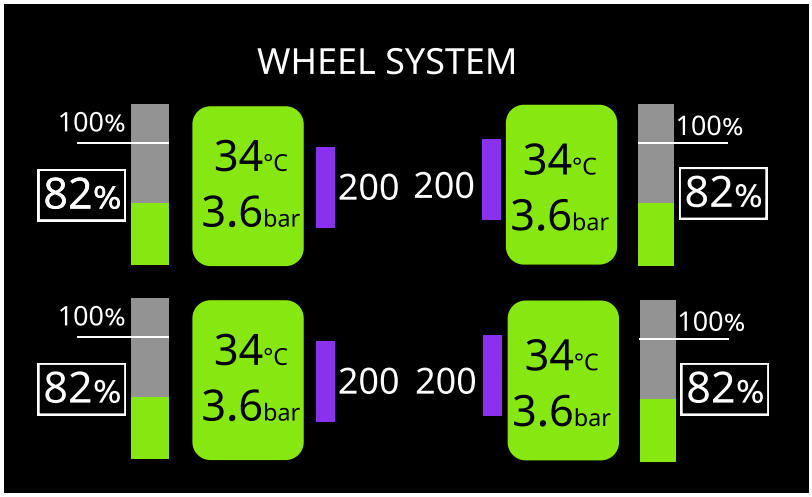


Figure 5.3: Part of the dashboard where the wheel system is displayed.

The wheel system part of the dashboard represents all the crucial information about the tyres, brakes, and dampers. In the middle, the purple bars represent the brake temperature; the bar changes colour according to the temperature, which is also represented by the number next to it. The big green parts are the wheels; this part also changes colour according to the temperature. In the wheel, the temperature and the pressure are also represented by numbers. The last part of the wheel system is the damper; the bar graph next to the wheel represents how much the damper has been compressed. The white line on the damper bar represents the value to which the damper should come at most; this way, the mechanic can see if the damper is being compressed too much, which can reveal possible damage or malfunction.

Fuel

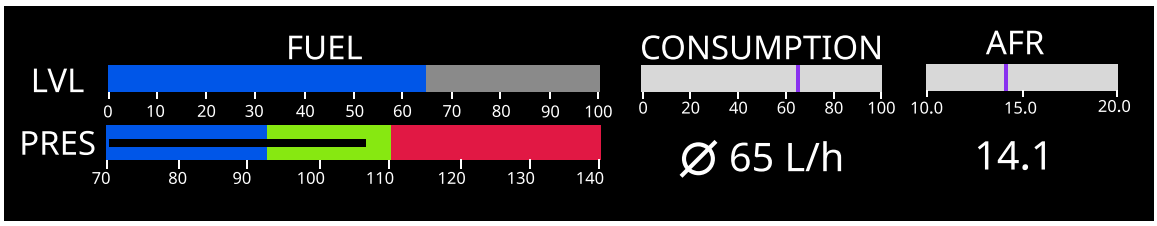


Figure 5.4: Part of the dashboard where information about fuel are displayed.

This section of the dashboard includes information about fuel. On the left side, there are two graphs. The one on the top represents the amount of fuel remaining in the tank, usually on the dashboard it's represented by a gauge, but I chose to use a simple bar graph to save space. The one on the bottom is a bullet graph representing the pressure in the fueling system. The next information represented in this section is the consumption of the fuel. The first graph represents the instantaneous fuel consumption, and at the bottom,

the average fuel consumption is represented by a simple number. The last part of this section is the air-to-fuel ratio, which represents the richness of the fuel being injected into the engine. There is a graph to see where the index is on the scale, and under it is a number representation of the same value.

Turbocharger

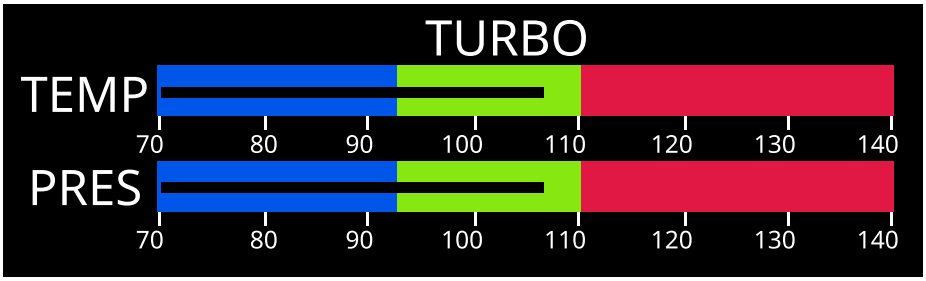


Figure 5.5: Part of the dashboard where information about turbo is displayed.

This is a representation of the state of the turbocharger. Two bullet graphs represent the temperature and the pressure that the turbo is pushing. Both visualizations have the same style of bullet graph in order to provide the range in which the values should be. The turbo's pressure and temperature are positioned directly beneath each other, allowing for quicker verification that both values are within the acceptable range.

Battery

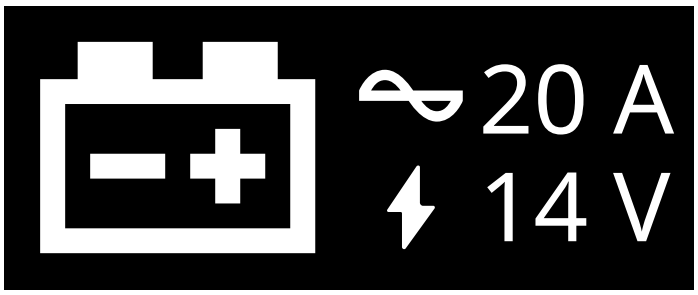


Figure 5.6: Part of the dashboard where information about battery is displayed.

The last graphic part of this dashboard is data that represents the current and voltage in the battery. The information is only displayed as numbers because the value doesn't change as much and is mainly used to check if it's still the same. However, it is good to know if the values are maintained. If the values drop to critical numbers, the color coding of the numbers will change to let the user know there is a problem with the system.

5.2 Second design

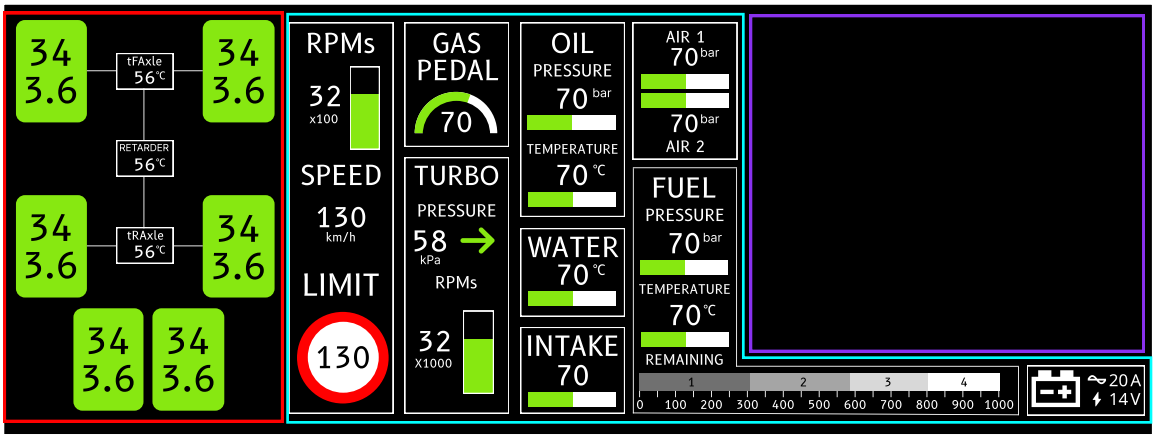


Figure 5.7: Final design of the dashboard.

This second design, which can be seen in Figure 5.7, was based on recommendations and requirements of Ing. David Svída Phd. It is also made with a focus on fast perception and not having any unwanted information that could fill the whole display without any purpose. The design has changed a lot since the first one, that's because after consultations with Ing. David Svída, Phd. I could understand the topic better and could focus the design more on what fits the use best.

The dashboard is divided into three parts, as seen in Figure 5.7. The first part in the red box is designated for the most essential information about the car, which includes the tyres, differentials, and also the retarder[1]. The second part in the blue box is a part of the dashboard dedicated to all the values that the mechanic needs to monitor the vital functions of the vehicle. This whole part is built from info tiles, which are all the white boxes that fill this whole part of the dashboard. And the last part of the dashboard is dedicated to error messages.

Essential info

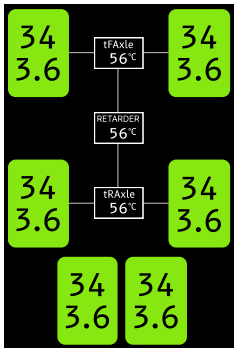


Figure 5.8: Part of the dashboard with the most essential information.

[1] A retarder helps the truck slow down, especially on long downhill slopes, without overheating or wearing out the normal brakes.

This part of the dashboard contains information about the temperatures and pressures in tires, differentials, and the retarder. The tire component is color-coded and signals the pressure in the tire. If the pressure is more than one bar, the tire is green. When the pressure drops below the one bar limit, the tire turns yellow to signal the mechanic that the tire is running low on pressure and that he should refill the tire. When the pressure drops below 0.5 bar, the tire turns red, which signals that the tire is really low and that there could be a puncture in the tire. Four tires are currently on the vehicle, and the bottom two are reserve tires that are on the back of the vehicle.

The middle parts between the tires are dedicated to the front and back differentials, and the middle one is for the retarder. These components ensure the correct power distribution for the front and back axles, which is why they are essential parts of the dashboard. These components are also color-coded, which means that the whole block turns red if the temperature goes out of a predefined range.

Vehicle Status Panel

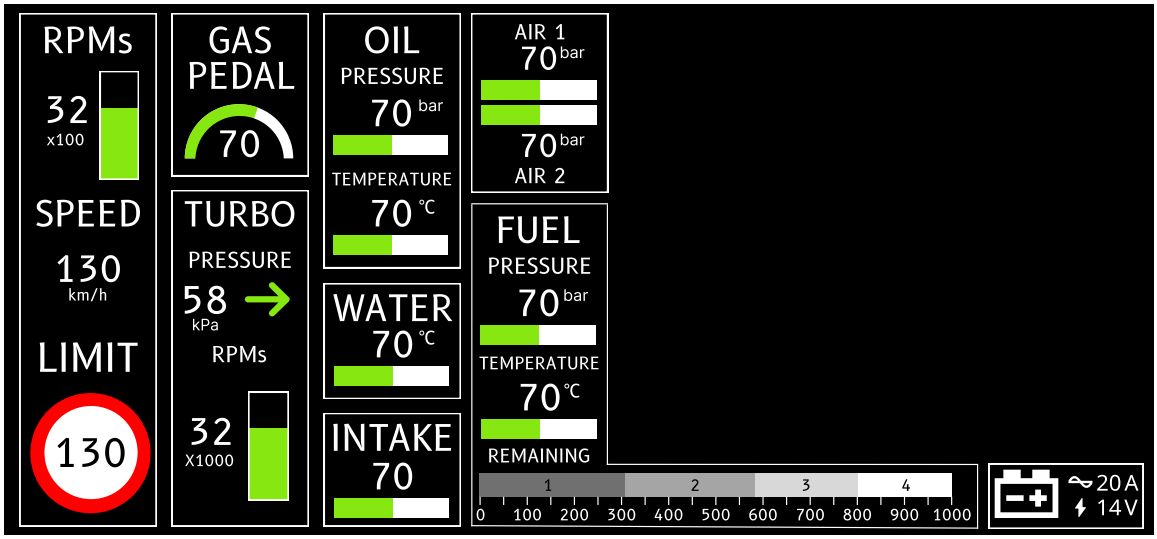


Figure 5.9: Part of the dashboard with information about almost all of the systems in the vehicle.

In this part of the dashboard, all of the vehicle status information is displayed. It is divided into nine boxes, where each box is dedicated to a different part or system in the vehicle. The first box is dedicated to information about the speed of the car. It has basic information, which contains RPMs, speed, and the speed limit of the car currently set. The second box is the gas pedal, which shows how much the pedal is pressed. The third box is related to the turbocharger, pressure, and RPMs of the turbocharger are displayed here, along with the arrow, which shows the trend of the maximum boost. The arrow that is available here is here so that the mechanic could analyze if the turbo maximum pressure is stable or dropping. This can help the mechanic to verify that the turbo is working as it should. The fourth box in this section is dedicated to oil. The fifth and sixth parts of this section are about water and intake. These parts are the same and only display the temperature of the cooling system and the air intake. The seventh part of the section is dedicated to the braking systems. These should always be the same, which is why the two bars are

in the middle; it helps with comparing the two values. The eighth part of this section is information about the fuel system, including pressure, temperature, and remaining fuel. The remaining fuel is a stacked bar graph because the truck has four different tanks, and the stacked bar displays the distribution of fuel. The last part of this section is the battery voltage and current.

The bars that display the values in this part of the dashboard are not basic bars that display values in a set range. To ensure that something is always displayed in the bars, I made a nonlinear bar. This was made based on consultation with Ing. David Svída, Phd, after he suggested that when the vehicle is started, the values are not in the ideal range that is set for each value, and it could lead to misunderstandings. The bar graph is loading from zero to the start of the ideal range in the first quarter of the bar, and the remaining three quarters of the bar are designated to the ideal range. This could also help the mechanic to ensure that all the sensors around the vehicle are working and sending some values.

As can be seen in Figure 5.9, the bar graphs that are used to display the values are aligned under each other. Since the bars are designed so that the bar graphs are in the middle, when the value is in the ideal state, the alignment helps with checking if the values are in a relatively good range. If the mechanic looks at the dashboard and sees that the bars are aligned in the middle, he already knows that all of the values are good. If anything is going out of the range, that will lead to the bar being on one of the sides of the bar, and it could be easily identified.

Error messages

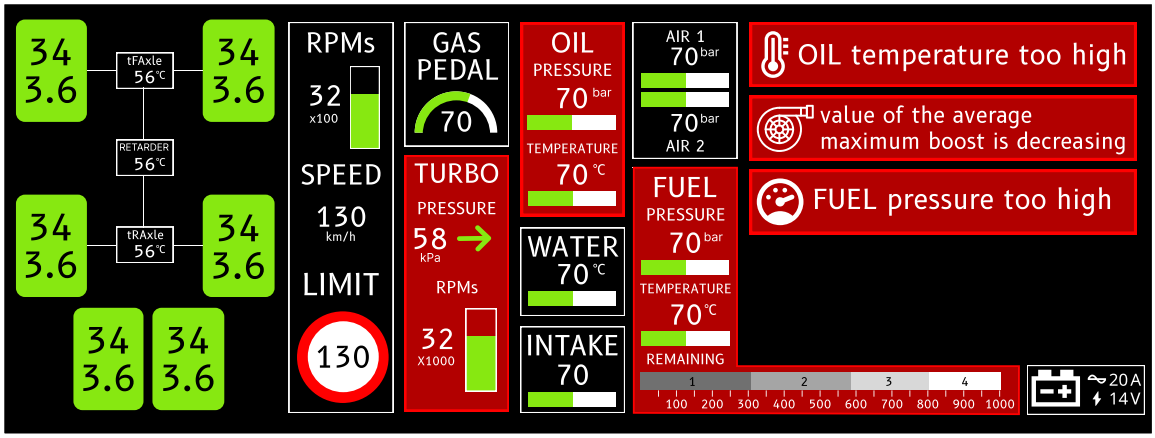


Figure 5.10: The dashboard with error messages displayed.

The last part of the dashboard is dedicated to errors. These errors are displayed as a box with an icon of what part of the vehicle the error message is related to, as well as information on what happened. When an error is displayed, the part of the dashboard that the error is related to turns red to enable a quicker visual connection between the error and the part of the dashboard it is related to. These error messages are raised based on predefined intervals in which the value should be. If the values go out of the range, the error message is shown and is displayed until the mechanic clicks on the error message. This ensures that the mechanic sees the problem and can acknowledge that he has seen it.

History graph

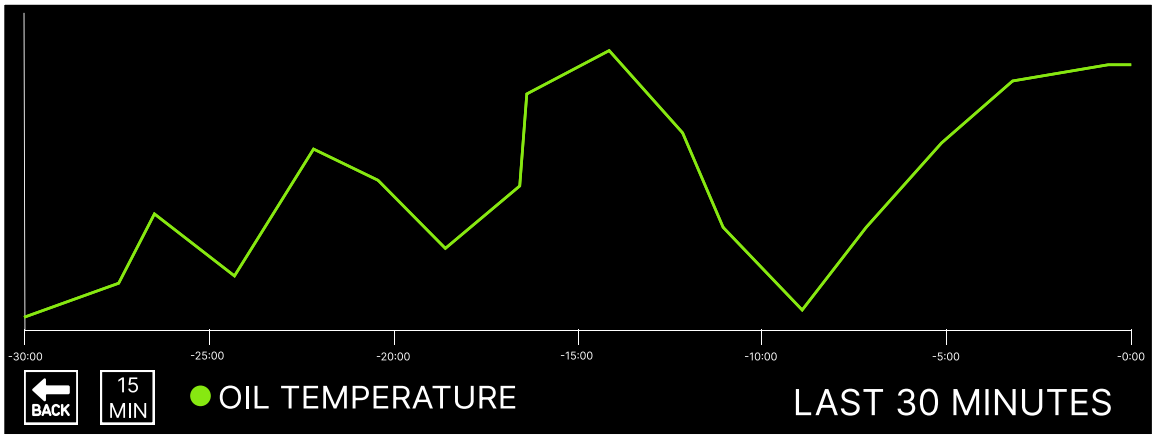


Figure 5.11: Sparkline displaying history of the value.

This graph, which can be seen in Figure 6.1, was designed to help the mechanic evaluate what could be the problem when anything happens. After clicking on the box that contains the desired value, the corresponding history graph will be displayed for the mechanic. The graph has two available details of the value. The first is 15 minutes and the second is 30 minutes. This is a simple graph that only displays the historical trend of the selected value, which is why the Y axis does not have any values on it. It will be displayed over the whole display; that's why there is also a back button, which will get the user back to the main dashboard.

5.3 Architecture of system

The outcome of this work will be a prototype in the form of a web application where the functionality of this dashboard will be demonstrated. Since this whole app will be just a prototype used to find the best possible solution, all of the visualization will happen on the frontend of this application. The app will be made using React and some React charting libraries.

In order to test the outcome of this work data simulation will be needed. This will be made with simulation recorded data from the app, and then parsing the data and displaying it in a set interval. For example, if the file for simulation will be in .csv format, the user will upload the file, the app will parse it, and then send the data row by row to the dashboard to make it look like it streams directly from the car.

Chapter 6

Implementation

As I already mentioned in Section 5.3, the main goal of this work was to make a prototype where I could demonstrate my proposal for the dashboard. The prototype is a web application where the frontend is made with React and TypeScript, and unlike the proposal stated, it also has a backend, which is made using two technologies: Express.js for the REST API and a WebSocket server for the live-data stream.

The communication between the client and the server of this app is visualized in Figure 6.1. The front-end passes the uploaded CSV through the REST API to the backend, where the real-time data is simulated and sent back to the front-end via the WebSocket connection. Other communication that happens between the two is requesting the history CSV files from the API when needed. The last is sending the simulation control messages via the WebSocket connection.

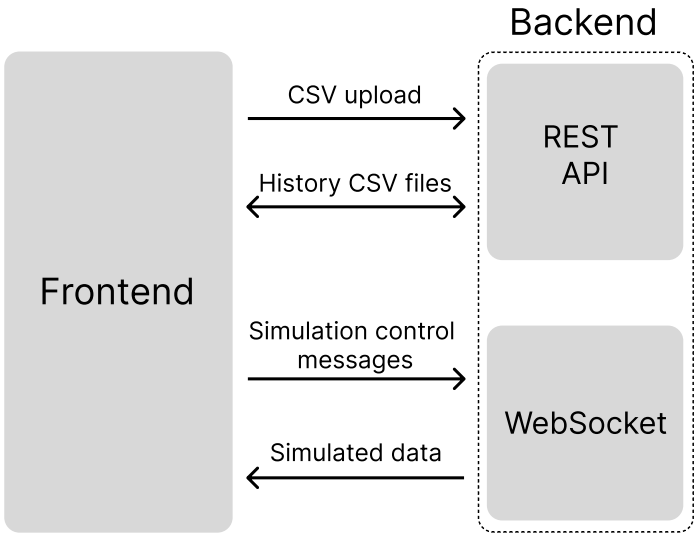


Figure 6.1: Communication diagram.

All endpoints and ports that are used for communication between the client and server can be seen in the table 6.1. In the endpoint `/api/data/:sensor`, the ":sensor" is replaced by the name of the sensor for which the file is requested. So if the frontend needs the record of oil temperature, the endpoint will be `/api/data/OILTEMP`.

Method	Endpoint / Port	Description
POST	`/upload-csv`	Uploads CSV data in JSON format to initialize the simulation.
GET	`/api/data/:sensor`	Retrieves historical CSV files for a specific sensor.
WebSocket connection	`4001`	Used for sending the control messages and the live data

Table 6.1: Backend communication endpoints aligned with the communication diagram.

In Figure 6.2, the whole app, ready for prototype testing, can be seen. There is a control panel used for running the simulation in the top left corner of the figure. Each button sends an appropriate message to the backend through the WebSocket connection. The CSV file used for the simulation is also uploaded through this control panel. After uploading, it is sent to the backend via a post request for further simulation. How the simulation is made will be explained in Section 6.2.

Figure 6.2: Web application made to present the prototype of the dashboard.

6.1 Dashboard

Some changes to the dashboard design were made during the implementation process. The changes were made based on preliminary testing sessions with Ing. David Svída, Phd. that happened during the implementation process. These changes will be explained in the following paragraphs.

Firstly, there were a few additions which can be seen in Figure 6.3. The first is the whole EGT2 info tile. It contains the temperature of the exhaust gases. The next addition is the analog water temperature sensor. This is crucial information that has to be always available, so in case one breaks, the other is still available. The last addition to the dashboard is the current time.

In order to provide a bigger space for the error messages on the right side of the box the the battery info tile from which the current was removed was moved to the top of the screen. Also, the remaining fuel bar was changed to a simpler and shorter visualization.

To ensure that the info tiles fit nicely into the middle part of the dashboard, some of them were moved to a different position.

Another change was that the trend arrow that was before, next to the turbo pressure, was replaced by a percentage. You can see this change in Figure 6.3. This percentage on the dashboard represents how the actual turbo pressure compares to the demanded pressure, which is also part of the dataset used for testing. The last change that was made was moving the engine RPMs to the same tile in which the gas pedal percentage is shown. These two values are highly related, and that's why Dr. Svída recommended putting them together.

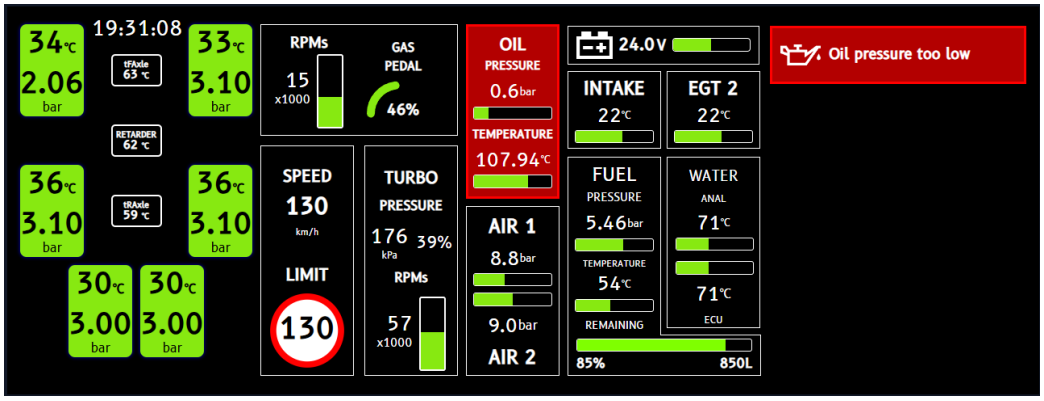


Figure 6.3: Implemented dashboard.

Context

To be able to share data across every component in this dashboard, I implemented a context provider. The implementation of the context can be found in the file `FileContext.tsx`. This context provides a centralized state management solution for handling the sensor data and control commands within the application. It is set up using React's Context API[1], which allows data (such as live data and error states) to be shared across components without the need for prop-drilling. In order to make the context available in any component wrapped inside the `FileProvider`, the custom hook `useFileData` is used; this hook consumes the whole context and makes it available for all the components. This hook has to be called in every component that wants to work with the context.

```
value={{
    liveData,
    sendControlCommand,
    errorMessages,
    dismissError,
    resetAllErrors,
    hasTurboError,
    hasFuelError,
    ...
}}
```

Listing 6.1: Contents of the shared context.

In Listing 6.1, all of the contents of the context can be seen. The `liveData` highlighted in red is a bit self-explanatory. It provides the live data streamed from the server. The

[1] https://react.dev/learn/passing-data-deeply-with-context

`sendControlCommand` is a function that is utilized in the control panel component. It is used for sending messages to the server side of this application. The yellow part of the context is used for error handling. Firstly, `errorMessages` is an array containing all the currently thrown errors. The next two functions highlighted in yellow are used to clear the currently thrown errors. The `hasError` flags highlighted in green are used to identify if any errors are present in the data shown in the info tile. There is one flag for each of the info tiles.

Error handling

The monitoring of the errors is handled in the file named `useErrorMessages.ts`. This hook is responsible for all the error generation and discovery. In this file, all of the ranges in which the values should be moving are defined.

When an error occurs, it has to be written into the `errorMessages` array available in the context. The messages are inserted as objects in the following form: `{ "error type", "error message" }`. The error type is later used during the generation to display an appropriate icon, and the error message contains information about what is wrong. When the object is added to the array, the new error message is automatically displayed in the list on the dashboard.

To change the appearance of the info tile connected to the error found, the appropriate `hasError` state in the context has to be changed to true. This automatically triggers the change of the info tile to the red visualization.

Component library

When implementing the dashboard, I focused on the scalability and easy configuration of the different parts of the dashboard. All of the info tile that are in the dashboard has their own dedicated React component. The only parameters that these components require are the displayed values, which are available in the context, the designated `hasError` flag, and lastly, to help with testing of different bar colors, the bar color is inserted into each info tile. In order to quickly change the colors of all the bars on the dashboard, the same variable is inserted into all of the components. To access the individual values from the `liveData` structure available in the context, the following command is used: `liveData."value name"`.

For visualizing the data in the info tiles, I made a component library that provides all the used visualizations. Most of these components are custom-made in order to meet the requirements of my work. These components are reusable to provide a possibility of usage in any further project. In the following paragraphs, all of these components will be presented and explained.

(a) (b) (c) (d)

Figure 6.4: Custom bar display media: a) number bar, b) inverted number bar, c) RPM, d) remaining fuel

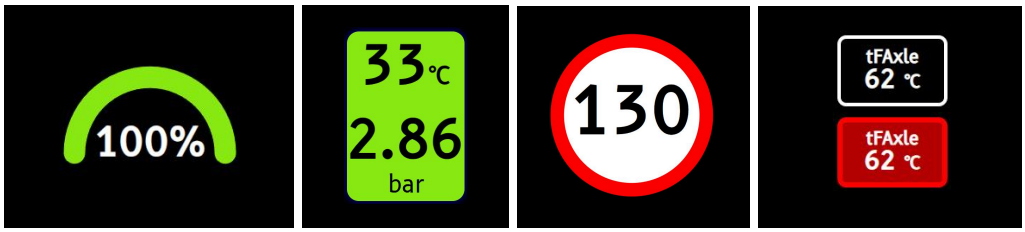
The first four components available in the library are all different modifications of a bar graph. The first two visualizations, number bar 6.4a and inverted number bar 6.4b, are implemented in a single component, and in order to change between these two, the prop "inverted" is used. However, there are more possible modifications that this component offers. The second modification is the selection between the linear and nonlinear bar. If the nonlinear bar is selected, the actual range of the bar is displayed on the last three quarters of the bar, and the first quarter is used to display the value when heating up. The last modification is whether the trend arrow should be displayed or not. The min and max values of the bar, along with the unit, are inserted into the component as parameters.

The trend arrow used in the number bar component is also implemented in a single component to add the possibility of reusing the trend arrow in different visualizations. The trend of the value displayed is calculated based on rolling averages. A rolling average is the average of a set of values within a moving window, where the window „rolls" forward as new values are added and old values are removed. Every time a new rolling average is calculated, it is compared to the previous one, and the percentage change between these two values is calculated. The equation used for this calculation can be seen in Equation 6.1. Then, based on the min and max thresholds, which are defined as parameters, the upward or downward arrow is displayed. The width of the rolling average can also be set as a parameter. This changes the frequency at which the values are compared, therefore it makes the arrow more or less responsible.

$$\text{Percentage Change} = \frac{\text{Current Rolling Average} - \text{Previous Rolling Average}}{\text{Previous Rolling Average}} \times 100 \quad (6.1)$$

The third visualization that can be seen in Figure 6.4c is a vertical bar made specifically to visualize the RPMs of different parts of the dashboard. The multiplier displayed under the value is set by a parameter, and also the min and max values of the whole bar. This bar is used to visualize the engine and the turbo RPMs in the dashboard.

The last of the bar visualizations that are made in this library is the remaining fuel which can be seen in Figure 6.4d. This is a simple bar visualizing the percentage of remaining fuel and also how much is left in the tank. This component only requires the min and max values of the bar and, of course, the value itself.



(a) (b) (c) (d)

Figure 6.5: Custom display media: a) percentage gauge, b) wheel, c) speed sign, d) temperature box

Another display medium that is made for these visualizations is the wheel component. This is the single wheel that is used to visualize the temperatures and the pressures in the tires. The background color is connected to the pressure being visualized in the given wheel. So if the pressure is under one bar but higher than half a bar, then the wheel is

yellow, and if the pressure drops below half a bar, the background turns red. Other than that, the tire is always green.

The gauge visualization is built using the ArcGauge from the `@progress/kendo-react-gauges`[2] library, which is part of the *KendoReact UI* component suite by *Progress*. This is a basic gauge that just requires the value that should be displayed and some settings on the visual side of the graph, and everything else is handled by the library.

The last two components in the library are the speed limit sign 6.5c and the temperature box 6.5d. The speed limit sign is a simple visualization that changes the speed limit based on the inserted value. The same with the temperature box, this is used when only the number is required for some value. However, the temperature box also offers the error visualization as can be seen in Figure 6.5d

The whole dashboard is then built in the file `Dashboard.tsx`, where the positioning and settings of the boxes are done. This file also implements the generation of the list with the error messages on the right side of the dashboard based on the `errorMessages` array from the context. For every message inside the `errorMessages` state, a new instance of the `ErrorMess` component is created.

`ErrorMess` component is represented by the red box on the right side of the dashboard, on the right side of the dashboard, where it displays the error message. This can be seen in Figure 6.3. Since the objects in the `errorMessages` state are in the format as was already explained, the first part of the object "error type" is used to choose which icon should be displayed in the error message box. The other part of the object from the `errorMessages` state is the message itself, which is written next to the icon in the error message box. The whole `ErrorMess` component is clickable, and after clicking on the error message, it will be removed, and also the connected box on the dashboard will be changed to the normal visualization without the red background. This is done by calling the function `dismissError()`, which was already explained before.

History graph

The final implemented version of the history graph is exactly as proposed, which could be seen in Figure 3.10. The graph is generated from history CSV files, which are being continuously updated with every new value. The graph provides two possible history windows to show. One is 15 minutes and the second is 30 minutes back. This is generated from the same history file, the only difference is that when the 15-minute visualization is chosen, only the first half of the file is filtered out from the array representing the whole 30 minutes.

The CSV files that are used for the history graph contain a series of rolling averages calculated from the live data. How long the rolling average is can be set differently for each value, so if a bigger detail is needed, it could be done. This rolling average is implemented to smooth out the line that will be generated from these values. All this rolling average calculation and also the file generation are handled on the backend. With every new value, the function `addSensorValue()` is called. This function adds the currently inserted value into an array from which the rolling average is calculated after the given time window. These averages are then logged into the files with the current timestamp. If the file with the history is already 30 minutes long, the oldest value is removed and a new one is added. Each value has its own file to ensure consistency in the logs when each value has a different time window for the rolling average.

[2] https://www.telerik.com/kendo-react-ui/components/gauges

After clicking on the info tile in the dashboard, the user will be redirected to the history graph. The boxes that have these features available are: oil, air, turbo, intake, EGT2, fuel, and water. After clicking on the box, an API request is sent to fetch the needed files for the generation. The API endpoint `/api/data/:sensor` is used for this communication, so when the frontend needs any file for any sensor, it simply fetches the file from this endpoint by replacing the ":sensor" with the name of the sensor.

After the history file is received on the frontend, the file is parsed and filled into an array from which the graph is generated. The array is filtered based on the range that should be displayed. Then, if the file is not long enough, for example, the dashboard was running only eight minutes, the remaining part of the array is padded with zeroes to ensure the scale on the dashboard is accurate. Then the whole array is inserted into the LineChart component from the `Recharts`[3] library, where it is rendered as a line representing the sensor's value over time, with a custom X-axis showing relative time (e.g., "-5:00") from the latest data point. If the clicked info tile contains two values, both are displayed in the graph as can be seen in Figure 6.6.

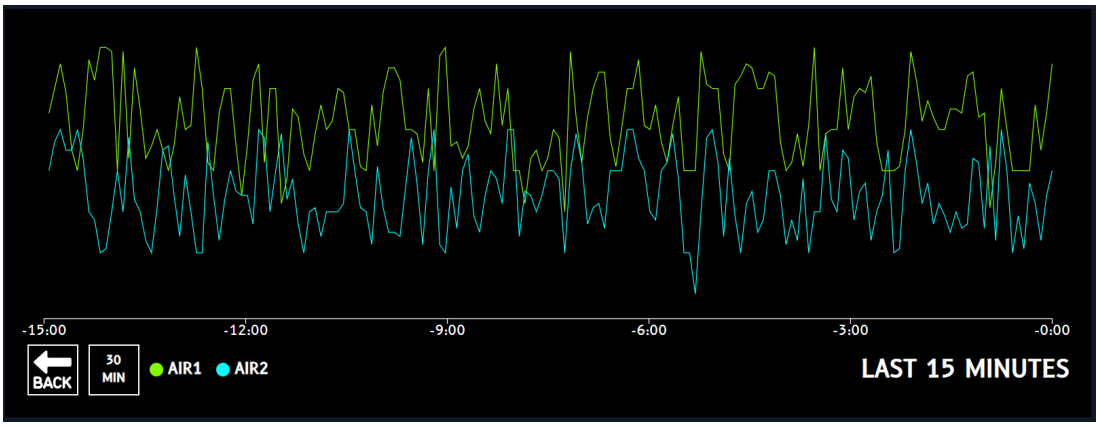


Figure 6.6: History graph displaying two values at once.

6.2 Data Simulation

As was already mentioned before, the whole simulation is run from a CSV file. This file needs to contain a header where the columns are set. Therefore, the values can be identified. The delimiter used in the file for separating the columns has to be ";" since in the Czech Republic, the "," is used as a decimal point, which is also the case in the file from Dr. Svída.

The file provided contains nine minutes of recorded data directly from the vehicle, sampled at a frequency of 10Hz, meaning the data was recorded every tenth of a second. The entire simulation is based on this data format.

[3] https://recharts.org/en-US

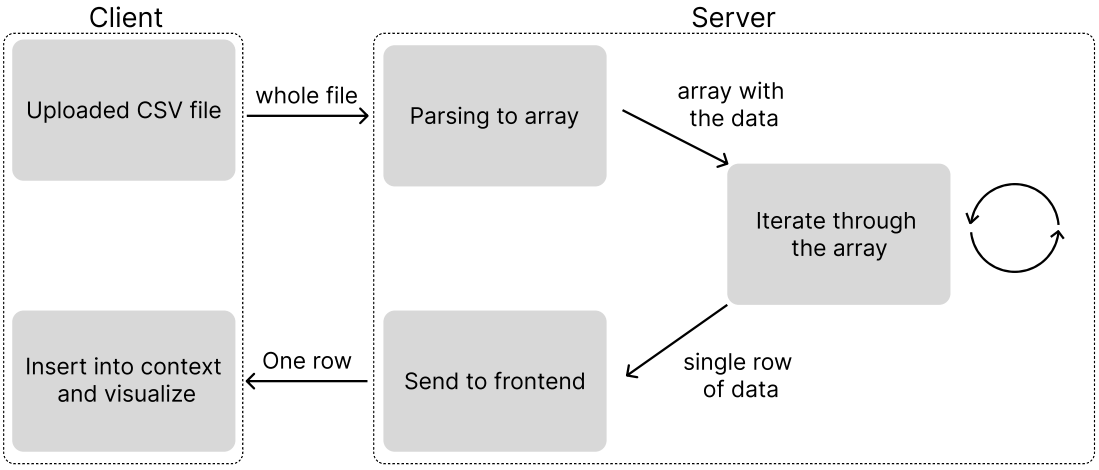


Figure 6.7: Diagram of the data flow during simulation.

As can be seen in Figure 6.7, when a user uploads the CSV file, it is sent straight to the backend via the `/upload-csv` endpoint. When the file arrives on the backend, it is parsed using the Papa.parse[4] library is a JavaScript object. The resulting array is then inserted into the `originalData` variable and is waiting for the simulation to start.

Now the server is waiting for the start message from the client. When it arrives, the server starts the iteration through the array of live data. A new row of data is sent to the client through the WebSocket connection every 100ms to simulate the 10Hz sampling frequency. This simulation runs until the whole array ends or until "pause" or "reset" message arrives. The pausing of the simulation is handled by the flag `isPaused`, which is controlled at the start of every iteration. If the reset message arrives the iterating is stopped as well and all the buffers used to track the history of the values are cleaned. Once the data arrives on the frontend, they are inserted into the `liveData` variable available in the context, which triggers all the components to reload with the new values.

[4]https://www.papaparse.com/

Chapter 7

Testing

The dashboard was tested with four consultants who are experienced in the world of the Dakar Rally. The first two consultants were already mentioned before, Ing. David Svída, Phd, and Ing. Martin Beran, engine designers for the trucks directly, and also a mechanics responsible for engine monitoring. The third person who had a look at my dashboard was Jan John, who attended the Dakar rally more than six times as a mechanic responsible for servicing the vehicle between stages, and who provided valuable insights based on hands-on experience with post-stage diagnostics and repairs. Filip Škrobánek is the next consultant who tested my dashboard. He is a race mechanic who attended several Dakar Rallyes as a mechanic directly in the car. And the last consultant I was testing with was also a mechanic, directly sitting in the car during the stages, and his name is Radim Kaplánek. All four of the consultants I was able to test with and evaluate the results of my work were highly experienced in the field I was making this dashboard for, which means that their opinions matter the most.

The first plans for testing were also to get the prototype into the truck directly; however, that could not happen because of the time schedules and the costs of the testing itself. The teams test their vehicle only a few times a year. Because of this, we were not able to execute the tests directly in the vehicle, which could have led to some interesting discoveries. But the insights that I got from the simulation testing are also very valuable information.

7.1 Testing scenario

Since the testing could not have been done in the real vehicle only the simulation, which was described in Section 6.2 was used for presentation of the dashboard. In order to show all the features, I made a demo.csv file. This file contains values that change in a way that demonstrates the features. For example, to show how the color coding of the wheels works, the pressure of the front left tyre is changing from three bars to a third of a bar. This way, all three states of the wheel are demonstrated. To show how the error messages work, the oil pressure is changing from three bars to a third of a bar. This will display the error message since the range of the oil pressure is set from 0.6 bar to seven bars. On this value, the trend arrow can also be demonstrated since the values are always changing in a significant amount. The last special value that this dataset has is the oil temperature. At the start of the file, the temperature goes from 20 to 100. This way, the nonlinear bar can be demonstrated. To demonstrate the functionality of the history graph, the simulation

was run in advance to ensure that sufficient data had been logged by the time the graph was displayed.

7.2 Testing results

The first tests with Dr. Svída and Ing. Beran were done to gather as much information as possible in order to provide the best possible solution. The base of the current dashboard, which could be seen in Section 5.2, was made mostly from the combination of the information he provided and the theoretical background that was explained in the first few chapters of this work. However, after making the prototype of the dashboard that was proposed in the section mentioned before, one more test was done with only Dr. Svída, where the changes explained in Section 6.1 were made. His opinion on the dashboard as a whole was that it could be highly usable in the environment it was made for, and that all of the features the dashboard provides could be a good contribution to reaching better results in the race.

Testing with Mr. John, Mr. Škrobánek, and Mr. Kaplánek had the same scenario. I presented the dashboard in the way that was explained in Section 7.1 and after they provided their thoughts and what could be improved or added.

First was Mr. John, who inspected the dashboard from a different point of view. As someone who is not in the car directly. There were a few changes he suggested. First, the speed of updating the information on the dashboard is a bit too fast, and it could distract the mechanic while trying to find some information. What he suggested was that the values that are rather stable and do not frequently change could be updated less frequently, only when they get closer to the end of the set range, the updating speed will be faster. For example, let's take the oil temperature into consideration. The working range of this value is from 90 to 120 °C. However, the temperature mostly fluctuates around 105 degrees. In this case, the frequency of the updates of the value could be only about two Hz. However, if the temperature gets near the end of the range, for example, five degrees from the start and the end, the updating frequency could get back to the 10Hz that it normally was. The second change that he proposed was that he would appreciate a button that would log an event into some file from which he could later, after the stage, look up the events and find what could have caused the issue that was happening at the logged time. However, his overall opinion on the dashboard was positive. He highlighted the clarity of the information and also that it provides all the information that could be needed for the race.

Mr. Kaplánek is currently using the dashboard that can be seen in Figure 4.4. His opinion was that my solution is a better way of providing clarity and also the fast perception of the values that are mostly needed. However, he also had a few comments on what could be improved in my provided solution. He noted that the connection between the dashboard element and the corresponding error message is well-designed, but expressed concern that, during a race, direct sunlight on the screen could cause him to overlook a newly displayed error. For this, he provided two possible solutions that could fix this possible issue. First is that the background of the whole dashboard could change to red or even start blinking red when the error appears. This will be much easier to see even in the bad lighting situations. The second opinion that he had was that the part of the display which is dedicated to the error messages is taking up a lot of space, and that during the race, not as many errors happen. What he would appreciate is that the part of the dashboard with all the information would stretch to the end of the display, and when an error happens, it will narrow down to the form it currently has, and the error messages are also displayed in the

same way. This could provide better visibility on the values, and the errors will draw bigger attention because of the bigger change on the display. Mr. Kaplánek also stated that he would like to have some audible signaling when the error happens so that he doesn't have to look at the dashboard all the time.

The last consultant who gave me feedback on the dashboard was Mr. Škrobánek. His first impression of the dashboard was that it had a clean and visually appealing design, and that the visualizations were easy to understand at first glance, which is essential in such environments. He complimented the display of the error messages and said that it would be easy for him to spot the error message. He also likes the fact that the color coding of the tires is changing based on the pressure in the tire, since that is the most crucial part of the car. However, he had a few things that he would like to add or change. The current solution that Mr. Škrobánek is using in the vehicle he is currently racing in has a bit different way of displaying the information.

In his solution, the dashboard is divided into three different screens that he can choose between. The first screen includes information about the tires, the second is dedicated to temperatures, and the last to the pressures. This way, if he finds any problem in the values, he can stay on the screen where he can monitor this and see better details of the values. However, the display he is using is only $8''$, which is $4''$ smaller than the one my dashboard is designed for. He said that my solution could be better, but that he would need to test it in real conditions. Another change that he suggested was that he would like to acknowledge the errors with an external button rather than clicking on the screen directly.

He would also like to have a possibility to turn off the sensor completely. This could be helpful in case some of the sensors break and are sending wrong values, which will lead to permanently displaying an error message. Also, when displaying the whole error message, he stated the same as Mr. Kaplánek, he would also like the whole display to blink when an error message is displayed. However, unlike Mr. Kaplánek, Mr. Škrobánek would want this to happen only when an error with oil, cooling water, and fuel happens, since these are the most essential values. The next change he proposed was to add an outside temperature value. This could help with identifying some problems connected to temperatures. The last addon that Mr. Škrobánek would like to have on his dashboard is more connected to his comfort rather than the performance of the car; however, that is also a very important thing. He would like to see the average speed of the car so that he could calculate the remaining time to the end of the stage.

7.3 Proposed extensions

I prepared a list of extensions that could help with improving this dashboard. All the extensions were made based on the testing results. The first proposed extension is that the whole dashboard will start blinking red if any error occurs. This could help with drawing attention to the dashboard and lower the chances that any error will be left unseen. The second extension I propose is stretching the dashboard across the whole display. This way, a better detail on the values could be provided. However, this also means that the visualization media will have to be redone to fit the width of the dashboard better, and with more space available, maybe provide more information. The third extension is that the error messages will be cleared by an external button. This could help in a race when the display sensitivity may be limited. Another button that should be added is a button to turn off the sensor completely. The reason for this extension was already explained before.

7.4 Conclusion of testing

The outcome of the testing indicates that the current solution I provided is largely effective and shows strong potential for real-world use in a racing environment. The visualization media that were chosen for the values are easy to understand and provide a suitable amount of detail. The changes everyone suggested were mostly based on personal preferences and what they are currently used to. However, there is one thing that was not mentioned in this testing, which is the Historical graph my dashboard provides. That is because the opinion on that was mostly the same for every individual. They stated that this historical sparkline that my dashboard has will probably not have any direct use. That's because during the race, there is no time to study the cause of the problem. And also, seeing just one value in the graph is not enough information to come up with a conclusion. However, Dr. Svída and Mr. Škrobánek said that if there were any graphs that combined values like the turbo boost and the gas pedal position, it could lead to some useful assumptions.

Chapter 8

Conclusion

The Dakar Rally is an extreme motorsport event that places unique demands on both vehicles and teams. To make sure that the vehicle is in the best possible state, detailed monitoring of the vehicle is needed. The vehicles are mostly monitored, controlled, and repaired between the different stages of the Dakar Rally. However, the vehicle's state also has to be monitored during the race. This is ensured by the onboard dashboards used by the team. The mechanic riding onboard has the most detailed dashboard in the vehicle. The visualizations on the dashboard have to be as intuitive and fast-perceptive as possible.

There are numerous existing solutions used to display the needed information. However, they often don't offer effective visualizations and are more focused on the information that the driver in the vehicle needs to know. That's why this work proposes a dashboard that contains all the needed information about the vehicle, error handling, and also a history graph. The data that is displayed on the dashboard has to be visualized in a way that is easiest to comprehend. That's why visualizations such as bars, gauges, and others are used. All the visualizations used in this dashboard are custom-made. They provide the much-needed simplicity and easy perception.

The full dashboard implemented in this work was made based on consultation with Ing. David Svída, Phd. and Ing. Martin Beran. They are both engine designers and mechanics with experience from several Dakar rallies. Some features which are designed to help with better monitoring of the vehicles state are: a nonlinear bar used in the dashboard to provide information about different values, trend arrow which enables monitoring of the evolution of the value, error message system which requires users acknowledgment to make sure the error has been seen, and also the history graph which displayes the historic evolution of the value.

All the proposed visualizations were implemented in a prototype of the dashboard in the form of the web application. This enables testing of the dashboard since it also implements a simulation, which takes the recorded data from the vehicle. This simulation was used during the testing with a wide range of people with direct experience from the Dakar rally. The results of the testing evaluate the proposed solution, where the people who tested the dashboard stated that the dashboard could have good usability in the real race. However, there are also a few proposed extensions which could possibly enhance the functionality of the dashboard, such as: more visible error indications, possibility to turn off a sensor, and a few more.

Bibliography

[1] AIM Technologies. *AIM MX Strada Series*. 2025. Available at: https://www.aimtechnologies.com/mx-strada-series/. Online.

[2] Akpakwu, G. A.; Silva, B. J.; Hancke, G. P. and Abu Mahfouz, A. M. A survey on 5G networks for the Internet of Things: Communication technologies and challenges. *IEEE Access*, 2018, vol. 6, p. 3619–3647. ISSN 2169-3536. Available at: https://doi.org/10.1109/ACCESS.2017.2779844.

[3] Bozdal, M.; Samie, M. and Jennions, I. A survey on CAN bus protocol: attacks, challenges, and potential solutions. In: *Proceedings of the 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE), Southend, UK, 16–17 August 2018*. Southend, UK: IEEE, 2018, p. 201–205. Available at: https://doi.org/10.1109/iCCECOME.2018.8658720.

[4] Campbell, S.; O'Mahony, N.; Krpalcova, L.; Riordan, D.; Walsh, J. et al. Sensor technology in autonomous vehicles: a review. In: *Proceedings of the 2018 29th Irish Signals and Systems Conference (ISSC), Belfast, United Kingdom, 21–22 June 2018*. Belfast, United Kingdom: IEEE, 2018, p. 1–4. ISBN 978-1-5386-6047-8. Available at: https://doi.org/10.1109/ISSC.2018.8585340.

[5] Cho, K.-T. and Shin, K. G. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In: *25th USENIX Security Symposium (USENIX Security 16), Austin, TX, August 10–12, 2016*. Austin, TX, USA: USENIX Association, 2016, p. 911–927. ISBN 978-1-931971-32-4. Available at: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cho.

[6] Ding, J.; Nemati, M.; Ranaweera, C. and Choi, J. IoT connectivity technologies and applications: A survey. *IEEE Access*, 2020, vol. 8, p. 67646–67673. ISSN 2169-3536. Available at: https://doi.org/10.1109/ACCESS.2020.2985932.

[7] ECUMaster. *ECUMaster ADU (Advanced Display Unit)*. 2025. Available at: https://www.ecumaster.com/products/adu/. Online.

[8] Farahani, S. *ZigBee Wireless Networks and Transceivers*. Oxford, UK: Newnes/Elsevier, 2008. ISBN 978-0-7506-8393-7. Available at: https://shop.elsevier.com/books/zigbee-wireless-networks-and-transceivers/farahani/978-0-7506-8393-7.

[9] Few, S. *Information Dashboard Design: The Effective Visual Communication of Data*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2006. 223 p. ISBN 978-0-596-10016-2.

[10] Fleming, W. J. Overview of automotive sensors. *IEEE Sensors Journal*, 2001, vol. 1, no. 4, p. 296–308. ISSN 1530-437X. Available at: https://doi.org/10.1109/7361.983469.

[11] González García, C.; Meana Llorián, D.; Pelayo García Bustelo, B. C. and Cueva Lovelle, J. M. A review about smart objects, sensors, and actuators. *International Journal of Interactive Multimedia and Artificial Intelligence*, 2017, vol. 4, no. 1, p. 7–10. Available at: https://doi.org/10.9781/ijimai.2017.431.

[12] Greengard, S. *The Internet of Things*. Cambridge, MA, USA: The MIT Press, 2015. The MIT Press Essential Knowledge Series. ISBN 978-0-262-52773-6.

[13] Gupta, M.; Benson, J.; Patwa, F. and Sandhu, R. Secure V2V and V2I communication in intelligent transportation using cloudlets. *IEEE Transactions on Services Computing*, 2022, vol. 15, no. 4, p. 1912–1925. Available at: https://doi.org/10.1109/TSC.2020.3025993.

[14] Hynek, J. *Impact of Subjective Visual Perception on Automatic Evaluation of Dashboard Design*. Brno, Czech Republic, 2019. Ph.D. thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Tomáš Hruška, C. prof. Available at: https://www.fit.vut.cz/study/phd-thesis/906/. Online.

[15] Jia, X.; Feng, Q.; Fan, T. and Lei, Q. RFID technology and its applications in Internet of Things (IoT). In: *Proceedings of the 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), Yichang, China, 21-23 April 2012*. Yichang, China: IEEE, 2012, p. 1282–1285. Available at: https://doi.org/10.1109/CECNet.2012.6201508.

[16] Li, S.; Xu, L. D. and Zhao, S. The internet of things: a survey. *Information Systems Frontiers*, 2015, vol. 17, no. 2, p. 243–259. Available at: https://doi.org/10.1007/s10796-014-9492-7.

[17] Lokman, S.-F.; Othman, A. T. and Abu Bakar, M.-H. Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review. *EURASIP Journal on Wireless Communications and Networking*, 2019, vol. 2019, no. 1, p. 1–17. Available at: https://doi.org/10.1186/s13638-019-1484-3.

[18] Madakam, S.; Ramaswamy, R. and Tripathi, S. Internet of Things (IoT): A Literature Review. *Journal of Computer and Communications*, 2015, vol. 3, no. 5, p. 164–173. Available at: https://doi.org/10.4236/jcc.2015.35021.

[19] Midway, S. R. Principles of effective data visualization. *Patterns*. Elsevier, 2020, vol. 1, no. 9, p. 100141. Available at: https://doi.org/10.1016/j.patter.2020.100141.

[20] Mikušová, M. and Janečková, V. Developing and implementing successful key performance indicators. *World Academy of Science, Engineering and Technology*, 2010, vol. 42, no. 6, p. 969–981. ISSN 2010-376X. Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2b0b809a7c5b6eff80a2aa4768b47154949e3251.

[21] Mohamed, K. S. *The Era of Internet of Things: Towards a Smart World*. Cham, Switzerland: Springer, 2019. ISBN 978-3-030-18132-1.

[22] MoTeC. *MoTeC C1212*. 2025. Available at: https://www.motec.com.au/products/C1212?catId=26. Online.

[23] Mountrouidou, X.; Billings, B. and Mejia Ricart, L. Not just another Internet of Things taxonomy: a method for validation of taxonomies. *Internet of Things*, 2019, vol. 6, p. 100049. ISSN 2542-6605. Available at: https://doi.org/10.1016/j.iot.2019.03.003.

[24] Pandharipande, A.; Cheng, C.-H.; Dauwels, J.; Gurbuz, S. Z.; Ibanez Guzman, J. et al. Sensing and machine learning for automotive perception: A review. *IEEE Sensors Journal*, 2023, vol. 23, no. 11, p. 11097–11115. Available at: https://doi.org/10.1109/JSEN.2023.3262134.

[25] Sadiku, M. N. O.; Shadare, A. E.; Musa, S. M.; Akujuobi, C. M. and Perry, R. Data visualization. *International Journal of Engineering Research and Advanced Technology*, 2016, vol. 2, no. 12, p. 11–16. ISSN 2454-6135. Available at: https://ijerat.com/index.php/ijerat/article/view/191.

[26] Samatas, G. G.; Moumgiakmas, S. S. and Papakostas, G. A. Predictive maintenance – bridging artificial intelligence and IoT. In: *Proceedings of the 2021 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 10–13 May 2021*. Seattle, WA, USA: IEEE, 2021, p. 413–419. Available at: https://doi.org/10.1109/AIIoT52608.2021.9454173.

[27] Sawant, P. R. and Mane, Y. B. Design and Development of On-Board Diagnostic (OBD) Device for Cars. In: *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 16-18 August 2018*. Pune, India: IEEE, 2018, p. 1–4. Available at: https://doi.org/10.1109/ICCUBEA.2018.8697833.

[28] Sergers, J. *Analysis Techniques for Racecar Data Acquisition*. Warrendale, PA, USA: SAE International, 2014. ISBN 9780768080810. Available at: https://books.google.cz/books?id=qXd0EAAAQBAJ. Online.

[29] Society, I. C. *IEEE Standard for Information Technology-Telecommunications and Information Exchange between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. Revision of IEEE Std 802.11-2016th ed. New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 2021. 1–7524 p. ISBN 978-1-5044-7284-5. Available at: https://ieeexplore.ieee.org/document/9363693.

[30] Song, Y.; Lin, J.; Tang, M. and Dong, S. An internet of energy things based on wireless LPWAN. *Engineering*, 2017, vol. 3, no. 4, p. 460–466. ISSN 2095-8099. Available at: https://doi.org/10.1016/J.ENG.2017.04.011.

[31] Sports, C. *How Data Analysis Transforms F1 Race Performance*. 2021. Available at: https://www.catapult.com/blog/f1-data-analysis-transforming-performance. Online.

[32] Todorovic, D. Gestalt principles. *Scholarpedia*, 2008, vol. 3, no. 12, p. 5345. Available at: http://www.scholarpedia.org/article/Gestalt_principles.

[33] Ventä, O. *Intelligent Products and Systems. Technology theme – Final report.* 635. Espoo, Finland: VTT Technical Research Centre of Finland, 2007. Available at: http://www.cs.hut.fi/~framling/Publications/intelligentproducts_final.pdf.

[34] Vongsingthong, S. and Smanchat, S. Internet of Things: A Review of Applications and Technologies. *Suranaree Journal of Science and Technology*, 2014, vol. 21, no. 4, p. 368–380. Available at: https://doi.org/10.14456/sjst.2014.38.

[35] Wang, R.; Yang, F.; Xia, H.; Huang, J.; Liu, C. et al. Internet of Things Technology in Motor Sports. In: *Proceedings of the 2023 International Conference on Intelligent Management and Software Engineering (IMSE), Rome, Italy, 8–10 September 2023.* Rome, Italy: IEEE, 2023, p. 153–156. Available at: https://doi.org/10.1109/IMSE61332.2023.00039.

[36] Ware, C. *Information Visualization: Perception for Design.* 4thth ed. Cambridge, MA, USA: Morgan Kaufmann, 2019. 560 p. ISBN 978-0-12-812875-6.

[37] Wexler, S.; Shaffer, J. and Cotgreave, A. *The Big Book of Dashboards: Visualizing Your Data Using Real-World Business Scenarios.* Hoboken, NJ, USA: John Wiley & Sons, 2017. 448 p. ISBN 978-1-119-28271-6.

[38] Zeadally, S.; Guerrero, J. and Contreras, J. A tutorial survey on vehicle-to-vehicle communications. *Telecommunication Systems.* Springer, 2020, vol. 73, no. 3, p. 469–489. Available at: https://doi.org/10.1007/s11235-019-00639-8.

[39] Zheng, H.; Paiva, A. R. and Gurciullo, C. S. Advancing from predictive maintenance to intelligent maintenance with AI and IIoT. *ArXiv preprint arXiv:2009.00351*, 2020. Available at: https://arxiv.org/abs/2009.00351.

Appendix A

Contents of the external storage media

- source/server – server part of the app
- source/client – client part of the app
  - /src – all components of the web
  - demo.csv – CSV file to show all the features of the dashboard
- This file in pdf.
- source-latex – folder with all needed source files for building this document