UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Juho Annunen

AN EXPERT EVALUATION OF A QT BASED DIGITAL TWIN INTERFACE

Master's Thesis
Degree Programme in Computer Science and Engineering
January 2024

Annunen J. (2024) An Expert Evaluation of a Qt Based Digital Twin Interface. University of Oulu, Degree Programme in Computer Science and Engineering, 61 p.

ABSTRACT

The popularity of the concept of a Digital Twin has been on the rise, partly because of the rise of Industry 4.0. Many areas, such as the aerospace industry, manufacturing industry, health care and transportation, are interested in the concept of Digital Twins. The interest has emerged because of the claimed benefits such as increased productivity, reduced production costs, sustainability enhancements, and reduction of environmental impacts. Some of the core technologies of the Industry 4.0 are, as well, important to Digital Twins.

In this project, a graphical user interface (GUI) software is developed to enable the user to start creating their own Digital Twin (DT) environments. The software includes a flexible custom Qt Modeling Language (QML) plugin that can be included in any Qt Quick application to enable sharing the data.

The software is presented to four experts, after which they have to answer a questionnaire consisting of questions about the presented software as well as DTs in general. The answers to the questionnaire are analyzed thematically and the results are confirmed via an agreement testing between two observers with Cohen's Kappa. The code of the software was analyzed statically, without running the code, with Axivion Suite, which is a static code analysis tool. This analysis provides some feedback about the quality of the code.

The answers to the questionnaire show the experts' perspective regarding DTs as well as their perspectives on the future of DTs. These results may be useful to future developers and researchers of DTs.

Keywords: Industry 4.0, Digital Twin, Qt, mixed methods, expert evaluation

Annunen J. (2024) Diplomityön teko-ohjeet. Oulun yliopisto, Tietotekniikan tutkinto-ohjelma, 61 s.

TIIVISTELMÄ

Digitaalisen kaksosen konseptin suosio on ollut nousussa Teollisuus 4.0:n suosion lisäännyttyä. Monet alat, kuten ilmailuteollisuus, tuotantoteollisuus, terveydenhuolto ja kuljetus ovat kiinnostuneita digitaalisista kaksosista. Kiinnostus on herännyt mahdollisten etujen takia, joita ovat esimerkiksi tuottavuuden paraneminen, tuotantokustannusten pieneneminen, kestävän kehityksen edistäminen ja ympäristövaikutusten pienentäminen. Jotkin Teollisuus 4.0:n tärkeimmistä teknologioista ovat tärkeitä myös digitaalisille kaksosille.

Tässä projektissa luotiin graafinen käyttöliittymäohjelmisto digitaalisten kaksosten ympäristöjen luonnin aloittamiseen. Ohjelmisto sisältää joustavan QML-lisäosan, jonka voi sisällyttää mihin tahansa Qt Quick -sovellukseen datan jakamista varten. Ohjelmisto esitettiin neljälle asiantuntijalle, minkä jälkeen heidän täytyi vastata kyselyyn, jossa oli kysymyksiä esitetystä ohjelmistosta sekä digitaalisista kaksosista yleisesti. Kyselyn vastaukset analysoitiin temaattisesti ja tulokset varmistettiin samanmielisyystestauksella kahden havainnoijan välillä. Ohjelmiston koodi analysoitiin staattisesti Axivion Suitella, joka on staattinen koodianalyysityökalu. Tämä analyysi antoi palautetta koodin laadusta.

Kyselyn vastaukset näyttävät asiantuntijoiden näkökulman digitaalisista kaksosista sekä niiden tulevaisuudesta. Nämä tulokset voivat olla hyödyllisiä tuleville digitaalisten kaksosten kehittäjille ja tutkijoille.

Keywords: Teollisuus 4.0, digitaalinen kaksonen, Qt, monimenetelmällinen tutkimus, asiantuntija-arviointi

TABLE OF CONTENTS

FOREWORD

First and foremost, I would especially like to thank Paula Alavesa from the University of Oulu for guiding me through this process, giving feedback, helping me with the analyses and reviewing this thesis. Thanks also to Vijayakumar Nanjappan from the University of Oulu for reviewing the thesis.

I would like to thank my supervisor Tommi Mänttäri for giving me feedback and new ideas for the thesis and the implementation. Big thanks also to Matti Paaso for helping me with the Axivion Suite.

My utmost gratitude to Kimmo Ollila for his efforts on the technical implementation of this project. Very special thanks to you for helping me get this project started in the first place, giving new implementation ideas and keeping them on a realistic level, taking part in and helping me with the programming and always pointing me to the right direction.

Lastly, I want to thank everyone else involved in the thesis project as well as my friends and family for the support and of course, thanks to Qt Group for the opportunity of doing the thesis here.

Oulu, January 25th, 2024

Juho Annunen

LIST OF ABBREVIATIONS AND SYMBOLS

3D	Three-Dimensional
5G	5th Generation mobile network
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
AWS	Amazon Web Services
CAD	Computer-Aided Design
CPS	Cyber-Physical Systems
DT	Digital Twin
DTDL	Digital Twins Definition Language
FW	Framework
GUI	Graphical User Interface
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IDE	Integrated Development Environment
IIoT	Industrial Internet of Things
IoT	Internet of Things
IPC	Inter-Process Communication
JSON-LD	JSON for Linking Data
MQTT	Message Queuing Telemetry Transport
NASA	National Aeronautics and Space Administration
OPC UA	Open Platform Communications Unified Architecture
PaaS	Platform as a Service
PLC	Programmable Logic Controller
PLM	Product Lifecycle Management
QML	Qt Modeling Language
QtRO	Qt Remote Objects
REST	Representational State Transfer
SaaS	Software as a Service
SDK	Software Development Kit
UI	User Interface
URL	Uniform Resource Locator
VR	Virtual Reality
XML	Extensible Markup Language
XR	Mixed Reality

1. INTRODUCTION

The popularity of the concept of Digital Twins is on the rise [1, 2]. The concept of accurate virtual replicas of physical objects has potential in a variety of areas. For example, a Digital Twin (DT) can be used in the aerospace industry, manufacturing, health care, transportation, energy, smart cities, agriculture and the building industry. The main benefits of using a DT in these applications are increased productivity, reduced production costs, and sustainability enhancements via services such as optimization and prediction. [3]

There are many levels of DTs; some are simpler, and some are very advanced. At their best, DTs can collect data from multiple sources, analyze it, make smart predictions based on the data and optimize systems [3, 4, 5]. For example, one of the most important elements of a DT in the aerospace industry is the ability to help predict failures of the aerospace systems [6]. A DT can control the whole system, and transfer data bidirectionally between the physical and the virtual models [7, 8].

The concept of a DT has been around for decades but has begun to gain more foothold only recently with the rise of Industry 4.0 and smart manufacturing where DTs are one of the key components [1, 3, 9, 10, 4]. Also, because modeling and simulation technologies have evolved, they are starting to be capable of handling DTs [1].

Industry 4.0 holds many key enabling technologies that are also important or fundamental to DTs, such as the Internet of Things (IoT), Industrial Internet of Things (IIoT), Cloud/Edge computing, Artificial Intelligence (AI), and Cyber-Physical Systems (CPS) [7]. IIoT network established in a smart factory may consist of a massive amount of sensor and actuator nodes feeding the DT continuously with real-time data from the factory [1].

DTs have matured from the time they were first introduced. The idea can be traced back to the 1960s when the replica was not even "digital", but an accurate physical replica of a physical system [4, 10]. In the early 2000s, the name Digital Twin was introduced in an educational context about product lifecycle management, and after that, the popularity of the concept has been rising, especially after 2012, when NASA and the US Air Force began to use DTs [4, 1, 2, 6].

There are already some companies using DTs, and they claim to have benefited from using them. Most benefits have to do with, for example, product development and prototyping cost and time reduction as well as reduction in environmental impacts. [11, 12, 13, 14, 15]

DTs also face some challenges and limitations. Some challenges are related to data privacy, security and interoperability. DTs generate and often have to access and process large amounts of sensitive data, which should be kept secure. DTs often require data integration from multiple data sources and platforms, which means that the data formats and protocols would have to be standardized. [3] Some studies have addressed this issue and proposed using open standards, such as Open Platform Communications Unified Architecture (OPC UA) [16, 17].

1.1. Method and Research Goals

The main research focus on this thesis is on qualitative data and its evaluation. The quantitative data is presented as well but due to the small sample size not analyzed as thoroughly. The quantitative data complements the qualitative data, providing a technical evaluation to the implementation in this thesis.

Expert evaluation method [18] was used and the collected data consisted of questionnaire answers, which were transcribed and analyzed with thematic analysis [19].

The goal of this explorative research was to study experts' views on DTs, and their future potential as well as receive feedback from the software that was implemented. The answers were analyzed using thematic analysis and confirmed by agreement testing with another observer. The agreements were measured by calucating Cohen's Kappa coefficients.

Technical analysis was also included in the evaluation; the code of the software was analyzed statically using Axivion Suite. With this tool, a set of code quality checking rules are selected for the analysis. The findings are presented and discussed after the analysis is executed. [20, 21]

1.2. Author's Contribution

The author of this thesis was responsible of planning the research and the implementation of the software. The experimental part included: arranging the software demo session, issuing the questionnaire, executing the thematic analysis, executing the agreement testing with another observer, calculating the Cohen's Kappa coefficients, and executing the static code analysis.

1.3. Structure of This Thesis

This thesis begins by presenting related work relevant to the theme of this thesis. An understanding of this background will also help in understanding the latter parts of the thesis. Then, the implementation part describes the software that was developed by describing multiple development phases so the reader can see how the software evolved from the early concept design to the proof-of-concept demo that was eventually presented to the experts. The experiments part describes what kind of research is done for this thesis and how it is executed. The results part presents and examines the results of both quantitative and qualitative data. The discussion part discusses the results and insights even further, points out some of the most interesting insights of the results, presents the limitations affecting this project as well as the further development ideas and plans for this project. The section also discusses what was learned and the project as a whole.

2. RELATED WORK

This chapter introduces the concept of Industry 4.0 and its related technologies. One of these technologies is a DT which is explained in more detail. This chapter presents a brief history of the concept, the structure of a DT, state of the art and some DT platforms. [22, 4]

2.1. Industry 4.0: The Technological Dependencies of DTs

Industry 4.0, which is also known as the fourth industrial revolution, is a widely used term that is used to describe the ongoing evolution of the manufacturing industry [22]. However, a fully accepted understanding of the term does not exist yet, and there has been criticism of the term itself, stating that there is no actual industrial revolution currently happening. It is said that Industry 4.0 does not have anything to do with new scientific breakthroughs like the previous ones and only contains merely upgrades to older technologies invented in the previous industrial revolution, such as controlling "old" technology with a modern smartphone using modern connections. [23]

Regardless of the criticism and confusion around the term, there are some fundamental components from which Industry 4.0 is considered to consist. All these technologies together enable what is known as smart manufacturing [24]. These include CPS, IoT, IIoT, cloud and edge computing, autonomous robotics, artificial intelligence and DTs [25, 26]. Many of these technologies already have applications in other areas, such as in the consumer industry. [22, 27]

IoT is a network of smart objects that communicate with each other [28]. Smart objects are embedded devices which are connected to the network [29]. Typically, these devices gather data using sensors and send it to other devices in the network so they can all cooperate to reach some common goals [30]. IoT has lots of use cases such as medical applications, logistics, industrial applications and smart home applications [29].

IIoT is referred to as the industrial applications of IoT. The main emphasis of IIoT is on smart manufacturing, which features interconnected smart machines that gather, analyze and interpret data via their sensors, actuators and other machinery. Based on this data, smart machines are capable of making intelligent decisions without human intervention. [31]

The purpose of CPS is to merge physical and virtual worlds together by combining computations and physical processes. Computers and networks may monitor and control physical processes via feedback loops, where the physical process may also affect the virtual process. [22, 32]

Cloud computing means sharing computation power and data storage capacity over the Internet [33]. Cloud services are scalable to users´ demands, which means that the resources of the cloud can be increased or decreased. The resources can be allocated by virtualization or containerization [34]. Some cloud services are also able to scale resources dynamically as on-demand. This is usually referred to as elasticity. [35]

Cloud services are usually divided into three different categories: infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS) and software-as-a-service (SaaS) with each providing a different level of abstraction [36, 37].

Cloud computing enables the data to be distributed across multiple locations across the network, but it is generally distributed across a few large datacenters. This kind of centralized model may introduce latency, especially in IoT systems where the IoT devices are not often located near the datacenters. [38]

The little more recent technology called edge computing is also widely used in Industry 4.0. Edge computing is similar to cloud computing, but in edge computing, the data is stored and processed as close as possible to where it is created to minimize latency. The data can be preprocessed by the local edge network, and there might not be a need to send all the data to distant datacenters at all. [38, 39]

2.1.1. Previous Industrial Revolutions

Prior to the current industrial revolution, there have been three industrial revolutions in the history of humanity.

The first industrial revolution took place in the second half of the 18th century. It was when steam engines were first introduced and used in factories to produce fabric, for example. This led to a major increase in productivity. [22, 27]

The second industrial revolution took place approximately 100 years later, in the 1870s. This introduced the use of electricity in factories. [22, 27]

The third industrial revolution, which is also known as the digital revolution, happened around the 1970s. This is when programmable logic controllers (PLCs) were introduced, and digital programming of automation systems started. [27]

2.2. Digital Twin (DT) Technology

Digital Twin (DT) technology aims to represent a physical object in a high-fidelity parallel virtual space in order to accurately simulate the object's behavior in the physical space and enable interactions between the two spaces [24, 4]. The two spaces should always be connected and synchronized to enable a real-time reflection of the physical space in the virtual space and the ability to observe, analyze and control the system in real-time [3, 40]. Both the physical and the virtual objects should be able to control the whole system. For example, if the virtual object changes its state, the change should be immediately reflected to the physical object. [7, 8]

Many of the Industry 4.0 technologies mentioned above are crucial to DTs. They can either be the technologies that enable the DT or improve it [9, 7].

The main difference between CPS and DT is that while CPS are considered the integration of physical and computational processes, a DT is the concept of an accurate and complete digital copy of a physical system [24]. DTs emphasise the virtual models that enable the one-to-one communication, whereas CPS focus on one-to-many communication. DT is dependent on the models that optimize, and predict the behavior of a system, whereas sensors and actuators are the core parts of a CPS [24].

DTs are used in a variety of areas, and a DT may have multiple use cases in each area. For example, in the building industry, a DT may be used for facility management, product lifecycle management, smart buildings or smart energy [41]. In the manufacturing industry, DTs may be used for monitoring the condition of devices, production lines and the whole factory in real-time. They can be used to collect data from the devices to optimize the quality and efficiency of the production and to analyze the operational data of the devices to maintain the devices predictively [3, 25].

2.2.1. The Parts of DT

A DT is considered to consist of three main parts: physical object, virtual object and the connections between them. The physical objects exist in the physical space and the virtual objects exist in the virtual space. [4, 42]

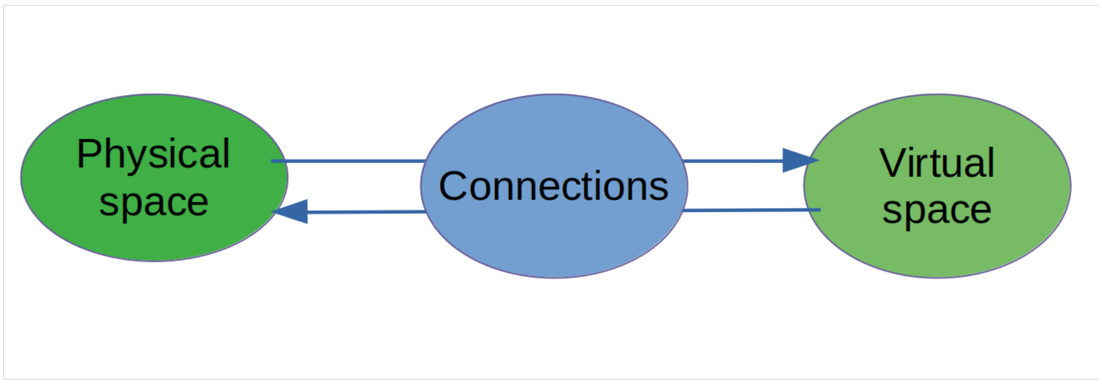


Figure 1. The parts of a Digital Twin (CC BY 4.0 Author)

The physical part of the DT is the physical object from which the virtual model is to be replicated. It can be anything from a single device, such as a robot arm, to a complete facility full of devices, such as a smart factory or even a complete city or a part of its infrastructure, such as transportation, energy distribution or water supply. With a constantly evolving, real-time data-gathering DT, these systems may be optimized. [3, 25, 13, 43, 44] The physical part can also be a process or an organization [5]. Based on their structure and function, the physical entities can be divided into three levels: unit level, system level and system of system level [2, 24, 5].

The virtual part of the DT is the replicated physical entity in the virtual space that accurately replicates the physical entity's geometry, properties and behavior [5]. It is usually represented graphically to the user as a high-fidelity three-dimensional (3D) model of the physical object, including all the physical properties. [25]

The virtual part consists of a set of models that enable the virtualized object to be represented virtually in three dimensions on the user's display. These models' job is to:

1. Reproduce the physical object's behavior
2. Operate autonomously in the virtual space to generate ideal behaviors in order to guide the physical counterpart's behavior
3. Predict future problems in the physical space and develop preventive actions

4. Validate the performance of the system

A DT may gather and combine data from both physical and virtual spaces. Some of the most advanced DTs use machine learning and artificial intelligence (AI) in order to operate continuously based on the gathered and combined data and are able to predict the future status of the system. [4, 5]

A DT needs connections between the virtual objects, the physical objects and the physical and virtual spaces. The physical objects can be connected with the corresponding virtual objects in order to achieve a stable closed loop for sensing and control. This ensures that the physical and virtual spaces are synchronized and consistent. [4] The data must be able to flow automatically in both directions to make a system a DT [7, 8].

If the data flows automatically, but only from the physical space to the virtual space, it is considered a Digital Shadow, not a DT. If the data is not flowing automatically at all, and all data updates are done manually in both directions, the system is considered to be a Digital Model, not a DT. [7, 8]

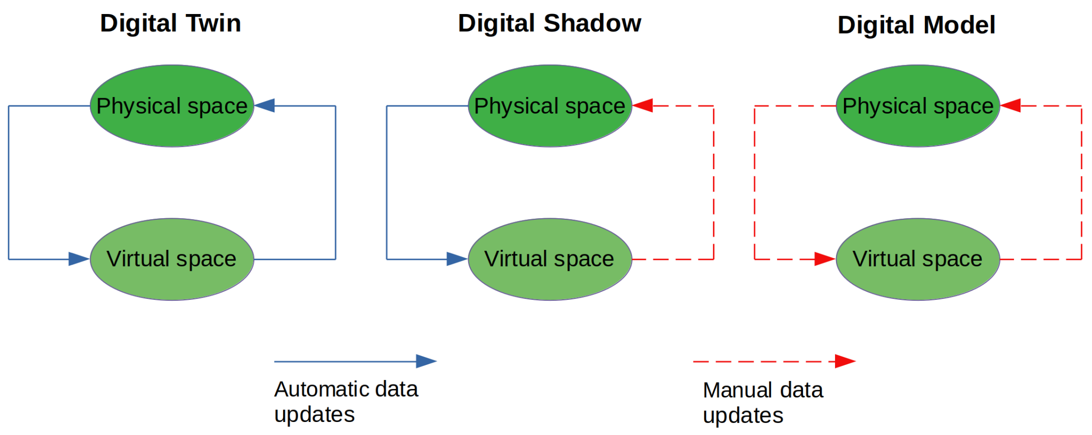


Figure 2. The difference between a Digital Twin, a Digital Shadow and a Digital Model (CC BY 4.0 Author)

In addition to the three parts described above and presented in the Figure 1 above, a five-dimensional DT model was proposed in 2018 by Fei Tao et al. [45] from Beihang University, adding data and services as separate parts to the original DT model [4].

A DT usually provides services for the end user, such as evaluation, optimization, prediction and validation. These services are usually used via a GUI, which are often black boxes from the user's point of view so that the user is able to use the services without any technical knowledge of how the services work. [4]

DTs generally handle lots of data. Data can be gathered from the physical space as well as generated by the virtual models in the virtual space, reflecting the simulation. It can include static attribute data, such as geographic location, spatial layout, etc., as well as dynamic condition data, such as changing environment, energy consumption, etc. [25, 5] In order to create a DT in the first place, lots of data must be measured and gathered from the physical space in order to create the high-fidelity virtual models.

This data may include shape, size, density, hardness, etc. These can be measured with existing measurement technologies such as laser, image recognition and nano-level precision measurement. Real-time data is then needed for the synchronization of the physical and virtual spaces. This includes for example, temperature, voltage, acceleration, pressure, etc. [5] This data can be gathered using IoT-connected sensors that gather data continuously from the physical space and transfer the data to the virtual space, which enables the virtual objects to replicate, for example, the movement of the physical objects. The virtual space may have virtual sensors that gather data from the virtual space and transfer it to the physical space [1, 25].

Data can be gathered and applied to physical and virtual spaces from the services, for example, prediction or optimization services. This is one of the most important features of DTs since it enables the DTs to predict and optimize in the first place. [4, 5]

Data can be brought into both the physical and the virtual space. If a physical product needs to be improved, the virtual counterpart must be tested and improved first before making any changes to the physical product [10].

2.2.2. How DTs Have Matured

DT concept can be traced back to the 1960s when the National Aeronautics and Space Administration (NASA) developed physical copies of their systems for the Apollo Program [10]. However, these systems cannot be considered as "Digital" Twins because neither of the systems were virtual. NASA developed two almost identical systems; one was launched into space, and the other stayed on Earth. This was done because the engineers wanted to mirror the conditions of the launched system. Later, NASA was able to replace their physical copies of the launched systems with virtual ones. [4]

The difference between a DT and a traditional simulation is that a simulation rarely has access to real-time updating data, while it is one of the most important features of a DT. Simulation relies on offline historical data and is mostly static until a new parameter, data, or other changes may be introduced in the simulation by hand. A traditional simulation cannot handle real-time updating of the data flow. In a DT, the virtual counterpart continuously gathers real-time data from its physical counterpart and its actual usage. The virtual counterpart may then analyze and optimize the data and send feedback back to the physical counterpart. Moreover, because of the modern connections that DTs generally use, this usually happens in almost real-time. [6, 46, 47]

Another difference is often associated with scale. While a simulation is usually focused on one particular process, a DT describes the whole environment, possibly including multiple processes and the connections between them. [46]

The concept of DT was presented by Michael Grieves in 2002 in his course about Product Lifecycle Management (PLM) [7, 4]. PLM is the process of managing a product and its data through its entire lifecycle, all the way from the design stage to the disposal stage [48]. Product's data is typically enhanced, modified and used throughout the lifecycle, and that is why perhaps the most important element of PLM is the creation, preservation and storage of the product's data in order to ensure fast and easy finding, distribution and reutilization of the data [49, 50].

Managing large amounts of product data can get difficult, if not impossible, very quickly if there is no virtual counterpart of the product involved. This is where DTs become useful since the DT collects and combines data continuously from the physical and virtual models. With DTs and IoT technology, all the data of a product may be available at a single address on the Internet. [4, 42]

2.2.3. DTs State of the Art

The Geminex by Metso (previously known as Metso Outotec) is a DT system used to optimize and predict mining and metallurgical operations, such as material enrichment. The Geminex's virtual counterpart is based on combined data from internal and external data sources. It uses AI for prediction and smart decisions. Metso claims that the main benefits of the Geminex are, for example: simulation without environmental or financial risks, maximizing the use of resources, minimizing the carbon footprint of the operations. [11, 12]

In June 2022, the French car manufacturer Renault Group claimed that their cars are built using DT technology. The virtual model tries to replicate the real physical car with as much detail as possible. They can then test the car's handling, aerodynamics, engine settings and safety-equipment without having to build a physical test prototype for each development stage. They also have a DT for their manufacturing process, including a virtual factory for optimization of the production quality. The group also uses a DT for vehicle monitoring throughout its lifecycle, from the vehicle's initial purchase to its disposal, including for example quality monitoring, customer feedback and information about repairs. Renault claims that with the use of a DT, they have managed to reduce a new vehicle's design time by a year. [13]

Framery has been able to adapt DT and Mixed Reality (XR) technologies in the product development of their sound-isolating booths. They have included the Varjo XR-3 Mixed Reality headset in their workflow. The booths have a real-sized virtual copy that can be used in testing and prototyping instead of making the changes to real physical prototypes or waiting for long periods of time for the new prototypes or parts to arrive. Changes made to the Computer-aided Design (CAD) files containing the components of the booth can now be reflected in the headset, visualising the booth immediately by using KeyShot and KeyVR [51, 52]. This enables a person to view the changes in the Mixed Reality environment immediately. Framery says that with this new workflow, they have been able to reduce the product design time by a month, skip a whole prototyping round and reduce prototyping costs. [14]

2.3. DT Platforms

Creating and managing a DT is a complex process that requires combining many technologies [5]. A DT requires software that handles, for example, modeling, simulation, real-time data streaming, interaction and data visualization [3, 5, 53]. There are numerous platforms for creating and controlling DTs such as GE Predix, Siemens Insights Hub (previously known as MindSphere), Microsoft Azure Digital Twins, Nvidia Omniverse and Amazon Web Services (AWS) IoT TwinMaker with each having their own advantages, differences and emphases [3, 54, 55, 56, 57, 58, 59, 15, 8, 60]. Many of these consider themselves as IoT or IIoT platforms, and they may have a wide amount of different kinds of use cases, but they can also provide the basis for a DT [61].

Microsoft provides a generic DT platform called Azure Digital Twins that is part of their Azure IoT services in their Azure cloud computing platform [62]. Azure Digital Twins is a PaaS that enables a user to develop and maintain a DT [62]. The service can be integrated with other Azure services, such as Azure Storage, Analytics, and Azure Machine Learning [62]. Azure Digital Twins uses Microsoft's own JSON for Linking Data (JSON-LD) based language called Digital Twins Definition Language (DTDL) for describing the data models for the DTs [63, 64, 60]. The service also has a tool for visualization of the data called Azure Digital Twins Explorer which can be used in conjunction with Azure Digital Twins, where a user can examine and modify their twin graph which is a graph of the structure of the DT containing information, such as the properties of the twins, their models and their relationships [65, 60].

One example of a customer using Microsoft Azure Digital Twins is GE Aviation, which uses the platform for modeling and observation of aircraft. They model the individual aircraft, track their asset performance, monitor their component usage and monitor maintenance history. GE Aviation claims that using the platform helps them make some critical maintenance decisions, reduce maintenance burden and maintenance costs as well as reduce the downtime or out-of-service time of an aircraft. [66]

Siemens Insights Hub is a broad IoT platform which can be used for DT purposes in a similar way to Azure Digital Twins by focusing on DT services and connections rather than the physical object modeling [5]. Insights Hub is a part of Siemens' Industrial Operations X platform. Insights Hub includes multiple tools to help manage the IoT environment, such as Asset Manager, Visual Flow Creator and Insights Hub Monitor. [55, 67]

The Asset Manager can be used for managing, creating and configuring the assets, which are defined by Siemens as digital representations of machines or automation systems with one or multiple automation units connected to Industrial IoT, which can be anything such as a pump, a motor, a robot, a car or a production line [67]. The Asset Manager can be used for actions such as defining data sources, making data connections to physical assets and sharing the assets with other tenants. The Insights Hub Monitor can be used to examine, visualize and analyze both the current and the historical data of the assets, view the shared assets, create and view IoT events and handle maintenance requests. It also allows users to create prediction models and offers anomaly detection services. The Visual Flow Creator can be used for creating visual representations of data processing workflows. [55, 67]

Insights Hub hosting environment can be used for hosting applications, but Insights Hub also allows users to integrate their externally hosted applications with the Insights Hub, providing access to Insights Hub Application Programming Interfaces (APIs). The application can then be managed via the Insights Hub launchpad. [55, 67]

Siemens provides multiple ways to connect user's hardware and software to the Insights Hub. User can develop their own connection based on, for example, Message

Queuing Telemetry Transport (MQTT), OPC UA or Siemens' own MindConnect software development kit (SDK), which supports multiple programming languages. Siemens also provides their own hardware with built-in connectivity solutions to the Insights Hub. [67, 68, 55, 69]

Nvidia's Omniverse platform is a scalable metaverse platform which is considered to be a multi-user collaboration environment that combines physical reality with virtual reality, often involving technologies such as Augmented Reality (AR), Virtual Reality (VR), XR, DT and Blockchain [57, 59, 70]. Omniverse provides real-time 3D and physics simulation, data visualization, as well as tools for developing applications and microservices. In addition to the appearance of the 3D objects, Omniverse also considers the physical characteristics of the simulated product, such as the properties of a material of an object in the virtual space [71, 72]. For example, Ericsson uses Omniverse to create DTs of cities where they can simulate their 5th generation mobile network (5G) accurately because the Omniverse also simulates the materials of the buildings where the 5G beams are reflected [72]. Omniverse has tools for application developers, such as a Python API as well as C++ SDKs [73]. Users can define connectors, which can be used to enable communication between external third-party applications and the Omniverse, which allows to import, export and synchronize, for example, 3D data [73]. The applications can be run via local machines, virtual machines, or they can be published to Omniverse Cloud Platform, where the end users can use it as a web application. [73]

AWS IoT TwinMaker is an IoT service that focuses on creating and managing DTs [58, 60]. Users can create their own models of, for example, physical devices, equipment and data, which can be transferred to the service via a Representational state transfer (REST) API or via a RESTful web service that is provided [60]. IoT TwinMaker lets users connect their own data sources to the system via connectors [74]. IoT TwinMaker has built-in connectors for first-party AWS data sources, such as the AWS IoT SiteWise service which is used to collect, store, organize and analyze device data, or the user can create their own connectors for third-party data sources [75, 76]. The platform supports a few common communication protocols, such as MQTT and Hypertext Transfer Protocol Secure (HTTPS) [77]. IoT TwinMaker has a tool called Scene Composer, which can be used to create 3D environments for the DT's virtual space from existing 3D and CAD models [60, 74, 75]. Real-time updating data can be integrated to the 3D environments, allowing the 3D environment to change according to the data updates. Real-world data collected by, for example, sensors, cameras or applications can also be visualized and monitored with the IoT TwinMaker. Users can create, for example, dashboards to help visualize the data and processes [60]. Knowledge graphs can be generated from the DT workspace, which represents the structure of the DT system containing information about the relationships between the DT resources [74]. The knowledge graphs can be integrated into 3D scenes, providing the ability to generate the graph from the 3D scene, containing the relations and information about the individual 3D models within the scene [74]. Finally, the IoT TwinMaker can be used to develop end-user DT web applications which can, for example, embed and combine some of the 3D scenes with the data dashboards. [74, 75]

3. IMPLEMENTATION

The purpose of this project was to develop a GUI software for Digital Twin environment orchestration called Digital Twin Orchestrator. The software can be used as a starting point for further development of DT environments.

3.1. Project Starting Point

This work demonstrates the system with a DT of a robot arm, meaning that the physical part of the DT would exist in the unit level [24]. The physical part of the DT was envisioned to be Alvin A.I. robot, a robot arm manufactured by Simua. The arm is laser cut from 3 mm plywood. The arm includes servo motors in the joints. It contains an ESP32 microcontroller [78]. [79] The full integration of the physical part of the DT was left out of this work in order to maintain the workload of the thesis manageable.

The virtual part of the DT is an application containing a 3D model of the robot arm 3. The application is a Qt application that demonstrates controlling the robot arm via an Android emulated controller application. These two are connected via a simulated Bluetooth connection. For this implementation, minor changes had to be made to the source code of the application as described below in the section Final proof-of-concept demo.

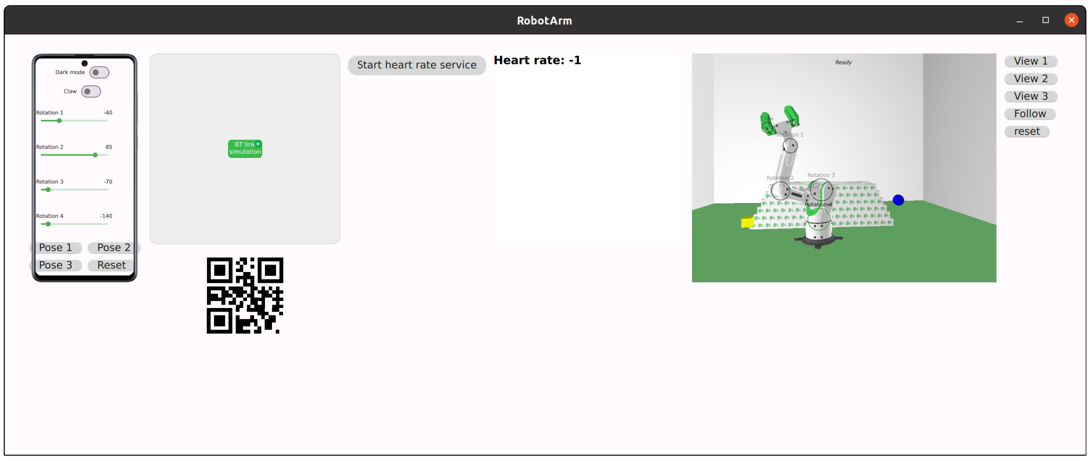
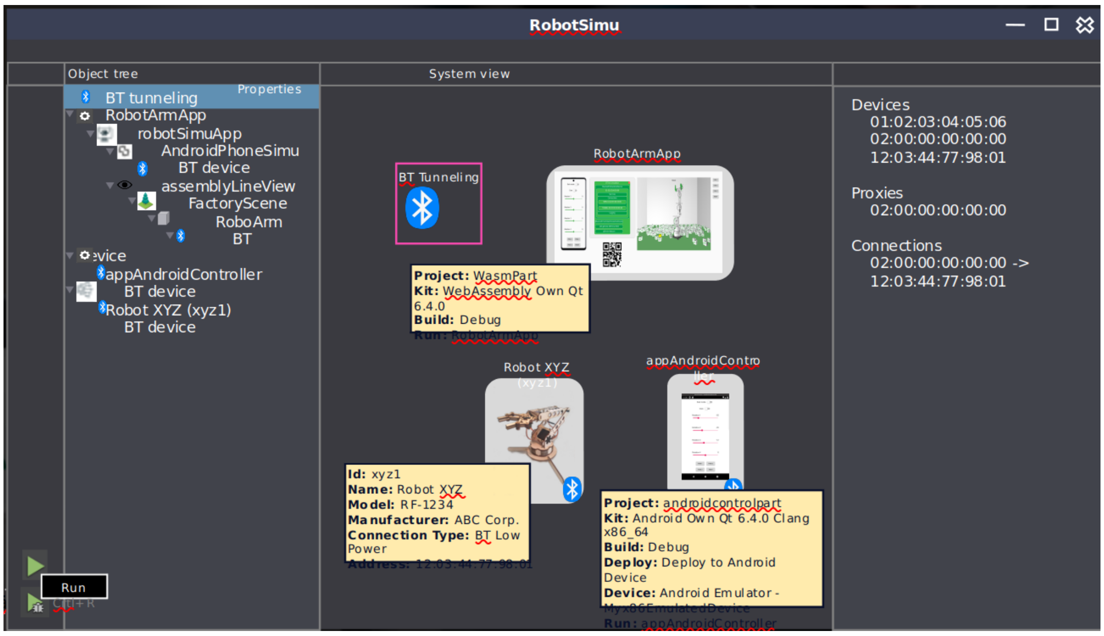


Figure 3. The robot arm for the demo application (CC BY 4.0 Qt Group)

The goal was that the user could control both the physical and the virtual arm using a physical or a virtual controller device. These devices could be connected via Bluetooth using the software's GUI (Figure 4).

3.2. Concept Design



Project: WasmPart
Kit: WebAssembly Own Qt 6.4.0
Build: Debug
Run: RobotArmApp

Id: xyz1
Name: Robot XYZ
Model: RF-1234
Manufacturer: ABC Corp.
Connection Type: BT Low Power
Address: 12:03:44:77:98:01

Project: androidcontrolpart
Kit: Android Own Qt 6.4.0 Clang x86_64
Build: Debug
Deploy: Deploy to Android Device
Device: Android Emulator - Myx86EmulatedDevice
Run: appAndroidController

Devices
01:02:03:04:05:06
02:00:00:00:00:00
12:03:44:77:98:01

Proxies
02:00:00:00:00:00

Connections
02:00:00:00:00:00 ->
12:03:44:77:98:01

Figure 4. The very first draft for the main UI, figure published with permission © Qt Group

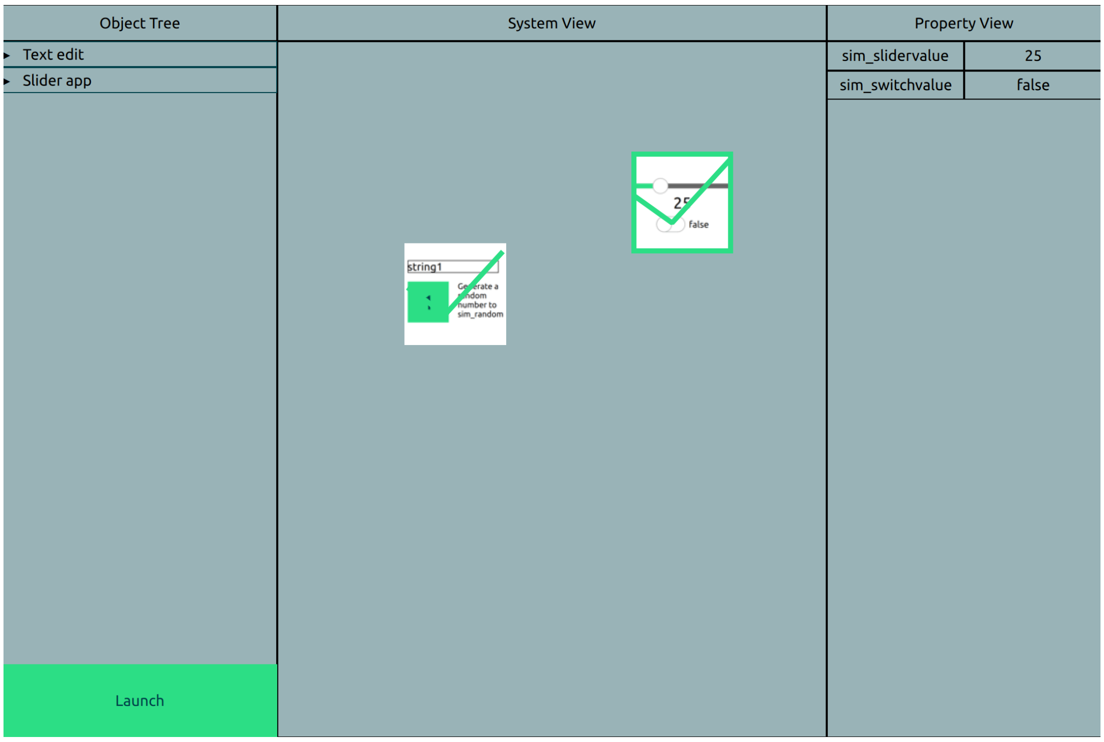
The draft UI consisted of three main parts. An object tree, a system view and a property view. The idea behind the object tree is that it would be generated automatically from the system view. It would show the hierarchy of the system view's components. The system view is the actual illustration of the DT components. The user should be able to build a DT according to his needs in the system view. The property view shows the inner properties of those applications that are created in the system view.

3.3. Technical Implementation and Platform

The software is developed using Qt framework and mainly Qt Quick. The programming languages used are C++ and QML (Qt Modeling Language) [80, 81, 82, 83]. The used build system is CMake [84, 85]. These tools are used inside the Qt Creator, which is Qt's own integrated development environment (IDE) [86].

3.3.1. Phase 1

The Figure 5 shows the first phase of the implementation. It has the three main parts described above. In the system view there are two applications instantiated. They can be seen on the same level in the object tree. The object tree is generated according to the hierarchy of the system view using a custom data model based on the QAbstractItemModel C++ class and its data is displayed using the TreeView QML component [87, 88].



sim_slidervalue	25
sim_switchvalue	false

Figure 5. The first implementation phase (CC BY 4.0 Qt Group)

The check-icons on top of the application icons indicate that the applications are currently running. They can be run using the launch button located in the bottom-left corner of the screen. When clicking this button, all the applications that are created in the system view and have their binary paths configured, are run as their own processes using QProcess C++ class [89].

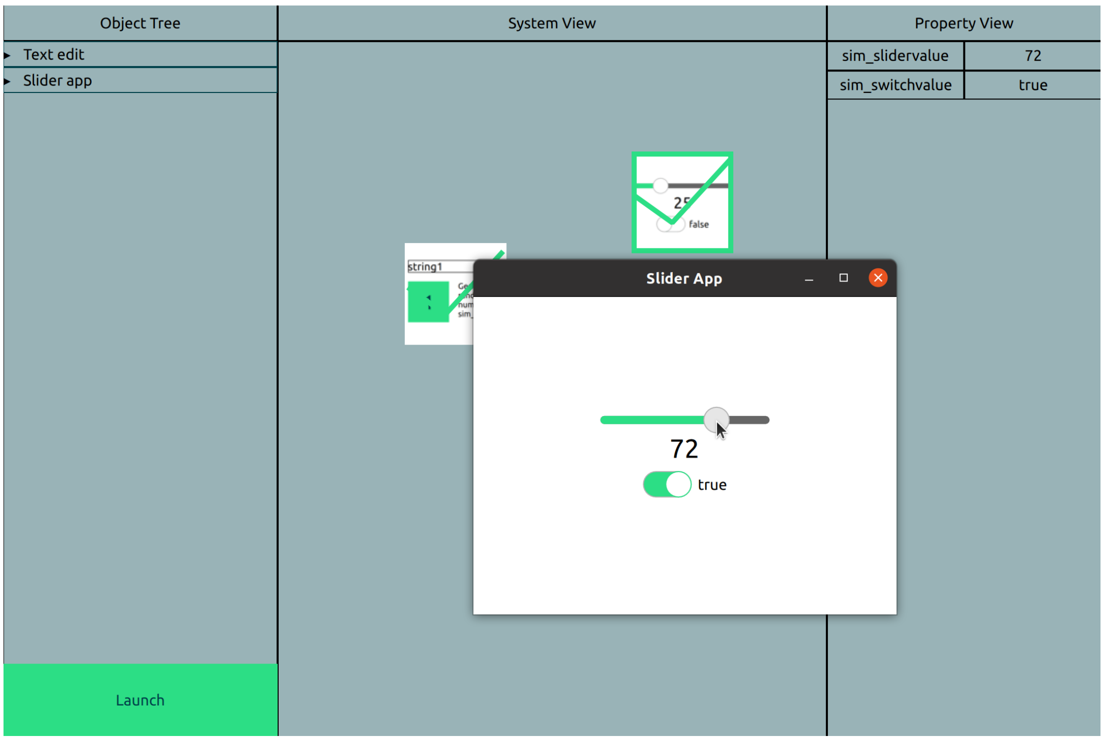
One of the applications can be selected at a time using a mouse click in the system view. When an application is selected, a green border line is drawn on the edge of the application icon and the application's inner properties are displayed in the property view.

These properties are updated in real-time when the property is modified in the application. This value is updated to the property view, where the user can see the value changing. The value updates are done using Qt's signal-slot mechanism and shared among nodes using Qt Remote Objects (QtRO) [90, 91]. QtRO is an Inter-Process Communication (IPC) module designed for Qt that utilizes Qt's own signal-slot mechanism for updating the values among nodes. QtRO can be used as a peer-to-peer network where each participant needs its own node. In this case, the external applications work as the host nodes, and the Digital Twin Orchestrator works as a client node. The user can also modify a value using the property view, making the value to be updated in the host node as well.

The external applications include a custom QML module called SimulationItem. This item is then instantiated in the QML application. The developer of the application may define a set of custom properties for the item and these properties are then scanned by the C++ class when the item is created in the QML application, and every property is then appended to a QStandardItemModel C++ member variable [92].

The application creates a host node with unique host URL (Uniform Resource Locator), connects to a common registry and enables the host node to provide remote access to the application's model. The client node (in this case, the Digital Twin Orchestrator) may then connect to any host node and obtain information about its model.

For example, in the Figure 6, in the Slider app example, the user can drag a slider to change a value inside the application. This value is then directly updated to the Orchestrator's property view via QtRO.



sim_slidervalue	72
sim_switchvalue	true

Figure 6. Example of the user changing a value in an example slider application (CC BY 4.0 Qt Group)

3.3.2. Phase 2

Phase 2 added a menu bar 7 to the application. Using the menu bar's File menu, a user can do multiple actions. Using the New action, a user can wipe out all simulation components from the System View, which also wipes out the Object Tree, creating an empty environment. Using the Save and Save As menus, a user can save the current environment to a file using Extensible Markup Language (XML) as the file format. From the Open menu, user can load these .xml files which then opens the saved environment. A Quit option quits the program.

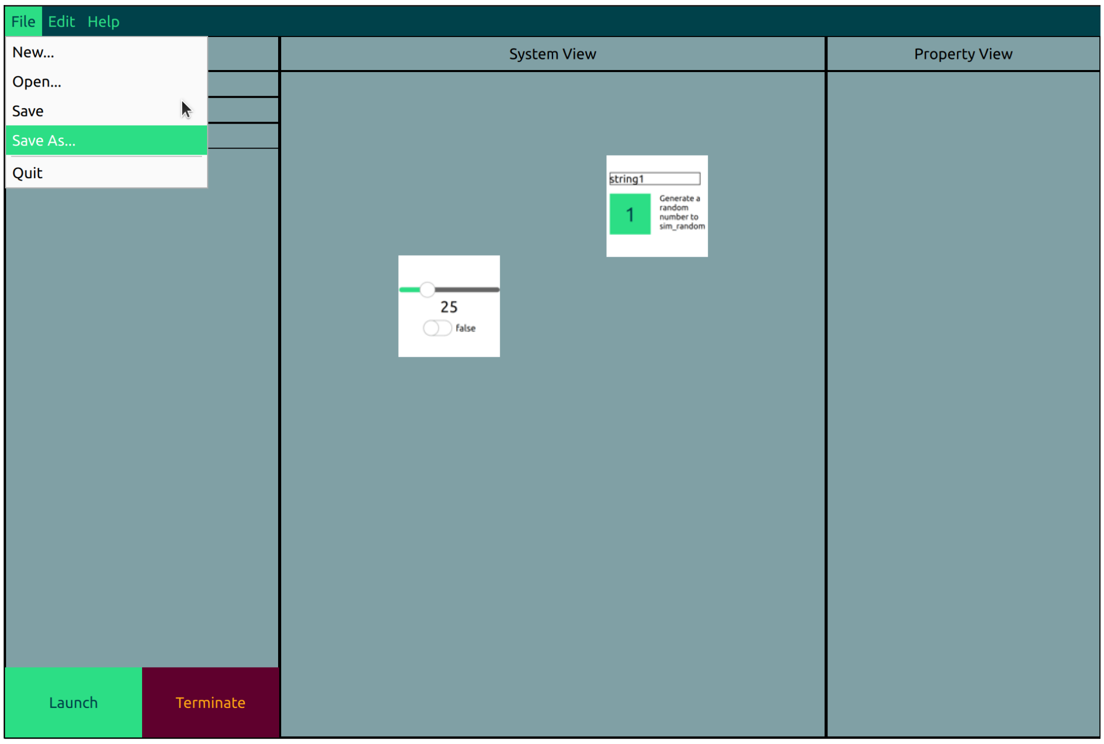


Figure 7. Menu bar's File menu (CC BY 4.0 Qt Group)

A Terminate button was also added next to the Launch button. This button terminates all applications that are currently in a running state.

A few context menus were also added. If the user right clicks anywhere on the system view, a context menu is opened on that spot. The menu has options to create demo apps from pre-defined presets and the option to create a custom application, meaning that the user can define the application's binary path and icon by themselves. Figure 8 demonstrates this menu.

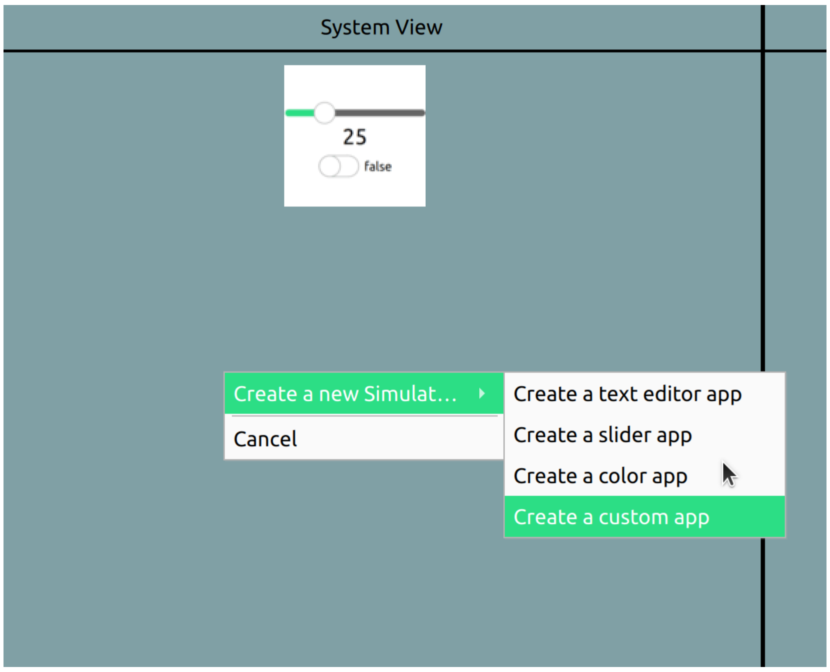


Figure 8. User attempting to create an application in the System View (CC BY 4.0 Qt Group)

Figure 9 shows that when an application is created, a user can right click on the application to open another context menu. This menu has options to delete the application from the system view, run or terminate the application and show the properties of the application. The option to show the properties of the application is only enabled if the application is currently running, and the option below changes whether the application is in a running state or not.

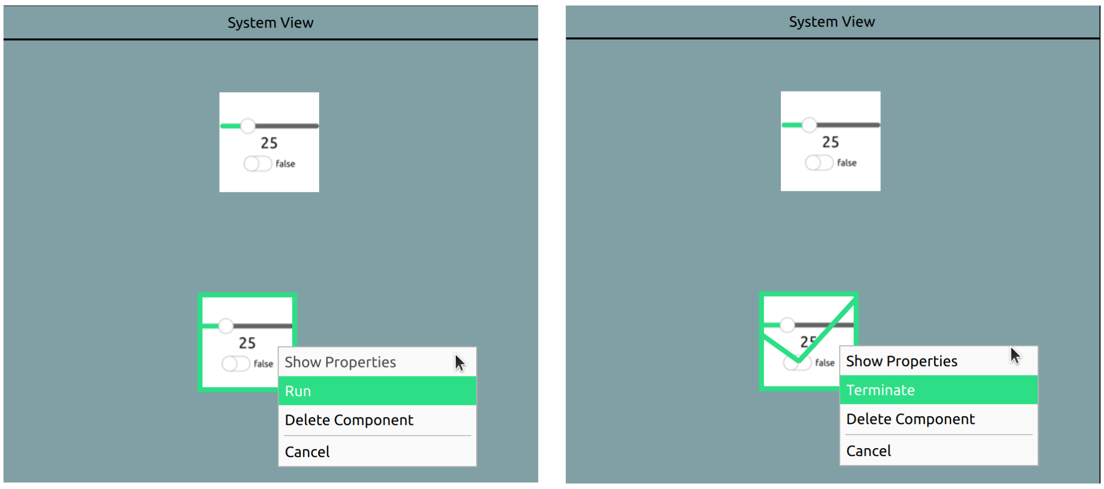


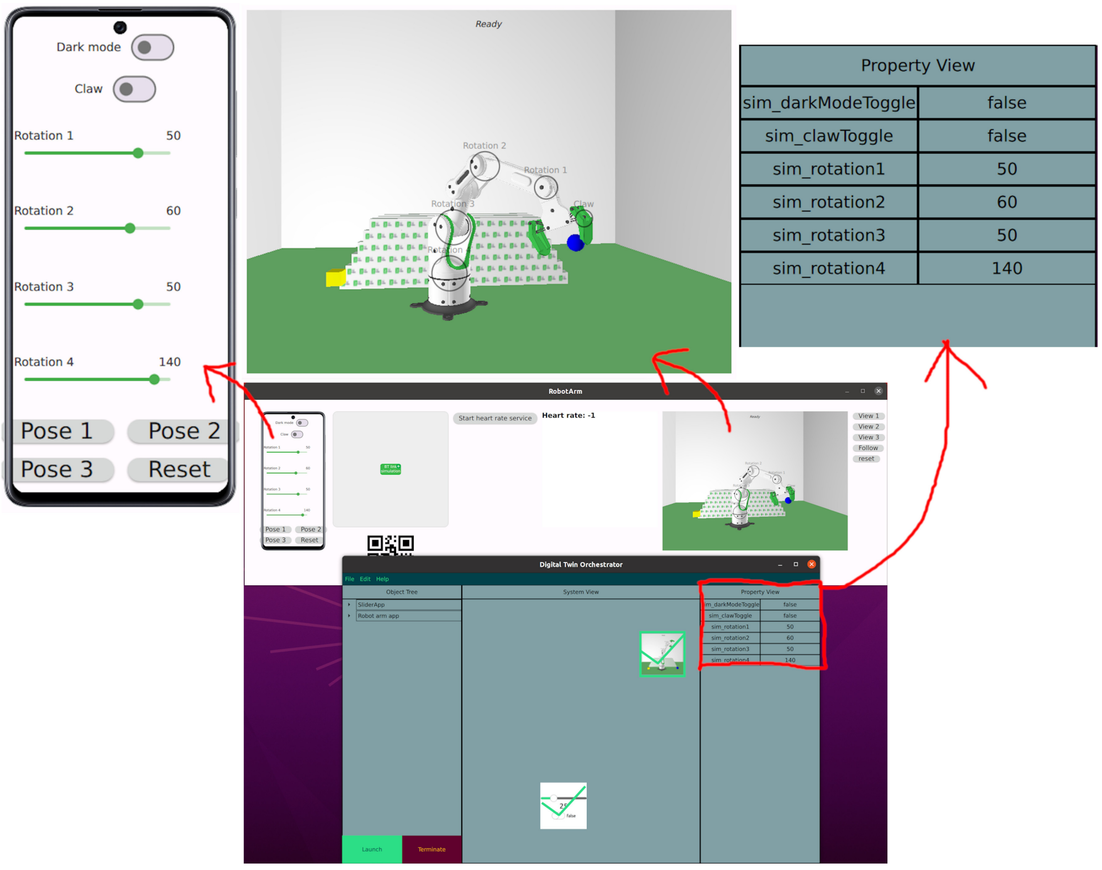
Figure 9. Context menu of an application that is not running (on the left) and context menu of a running application (on the right) (CC BY 4.0 Qt Group)

3.3.3. Final Proof-Of-Concept Demo

In the final version, a user can create an empty application, give it a path to a binary and give the application an icon so that it can be recognized immediately in the system view. With these steps, a user can launch any application via the Orchestrator, providing that the application has been built successfully before, and as a result, has a binary file which can be launched.

For this work, I had to modify the robot arm application that was mentioned in the beginning of this chapter, so that it includes the SimulationItem plugin. Then I could create a SimulationItem instance within the code and define variables to be shared to the Orchestrator. I bound these variables to the values of the robot arm joints [93]. As a result, the SimulationItem could now append these variables to its model and share the model with the Orchestrator, enabling bi-directional value updates with the actual application and the Orchestrator.

After the brief modifications, I built the robot arm application with Qt using CMake, gave the binary's path to the application using the object tree, and gave it an icon. Now the application is ready to be launched within the Orchestrator (Figure 10).



Property View	
sim_darkModeToggle	false
sim_clawToggle	false
sim_rotation1	50
sim_rotation2	60
sim_rotation3	50
sim_rotation4	140

Figure 10. User modifying joint values via the property view (CC BY 4.0 Qt Group)

4. EVALUATION

This thesis' evaluation was done with mixed method approach collecting both, qualitative and quantitative data. The qualitative data was collected in an expert evaluation [18]. The quantitative data addressed the technical functionality of the UI and was be collected using static code analysis, which means analyzing the program code for violations of rules without executing the program [21].

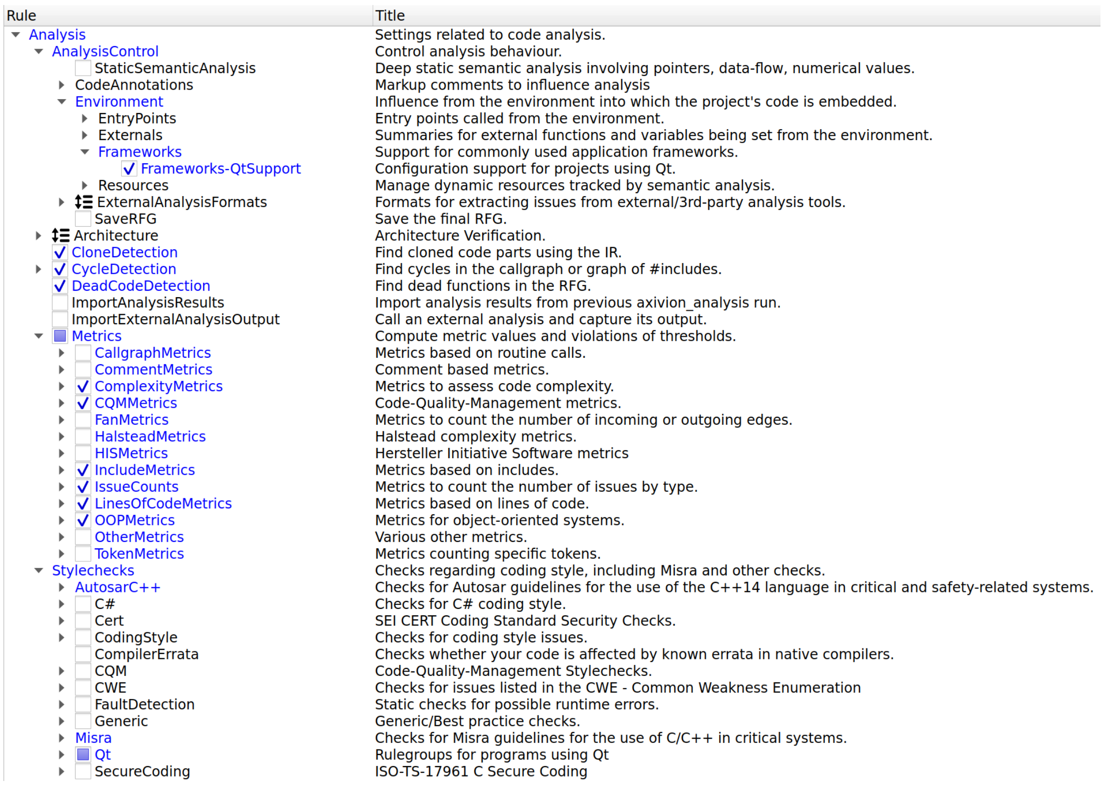
The expert evaluation began with a demo session presenting the Orchestrator followed by a questionnaire containing questions about DT technology and the Digital Twin Orchestrator. The experts answered a questionnaire and the answers were analyzed using thematic analysis and agreement testing with Cohen´s Kappa analysis [19, 94].

The quantitative evaluation was executed as static code analysis using Axivion Suite which is a code analysis tool including for example, static code analysis, architecture analysis and code smells detection. The static code analysis aims to find weaknesses from the code without running the program. Axivion includes various style checks, which check the code's compliance against desired coding guidelines. Axivion includes various predefined style checks but users can also add their own rules. Code smells that Axivion can detect include, for example, duplicated code, header include cycles, call cycles and unreachable code. [20]

4.1. Static Code Analysis Procedure

For the technical analysis the whole Digital Twin Orchestrator code base was analyzed with Axivion [20]. First, an Axivion Suite Dashboard must be setup on a server where the results of the analyses are stored and where the user can examine the results. For this project, the Axivion Suite Dashboard ran on a localhost server.

Before running the analysis, the analysis has to be configured to tell Axivion, which rules the user wants to include in the analysis. This can be done by editing Axivion's rule_config.json file, which can be opened for editing using Axivion's Project Configuration tool. There, a user can check all checkboxes that relate to the rules that should be included in the analysis. For this project, I chose to include, for example, cycle detection, dead code detection, OOP metrics and Qt stylechecks, which contain Qt-Autosar, Qt-Clazy and Qt-Generic. I feel that these were the most relevant for this kind of project at this point of development. Figure 11 shows all the rules that are included in the analysis. They are indicated by blue check marks or blue squares in checkboxes in the list.



Rule	Title
Analysis	Settings related to code analysis.
AnalysisControl	Control analysis behaviour.
StaticSemanticAnalysis	Deep static semantic analysis involving pointers, data-flow, numerical values.
CodeAnnotations	Markup comments to influence analysis
Environment	Influence from the environment into which the project's code is embedded.
EntryPoints	Entry points called from the environment.
Externals	Summaries for external functions and variables being set from the environment.
Frameworks	Support for commonly used application frameworks.
✓ Frameworks-QtSupport	Configuration support for projects using Qt.
Resources	Manage dynamic resources tracked by semantic analysis.
ExternalAnalysisFormats	Formats for extracting issues from external/3rd-party analysis tools.
SaveRFG	Save the final RFG.
Architecture	Architecture Verification.
✓ CloneDetection	Find cloned code parts using the IR.
✓ CycleDetection	Find cycles in the callgraph or graph of #includes.
✓ DeadCodeDetection	Find dead functions in the RFG.
ImportAnalysisResults	Import analysis results from previous axivion_analysis run.
ImportExternalAnalysisOutput	Call an external analysis and capture its output.
▨ Metrics	Compute metric values and violations of thresholds.
CallgraphMetrics	Metrics based on routine calls.
CommentMetrics	Comment based metrics.
✓ ComplexityMetrics	Metrics to assess code complexity.
✓ CQMMetrics	Code-Quality-Management metrics.
FanMetrics	Metrics to count the number of incoming or outgoing edges.
HalsteadMetrics	Halstead complexity metrics.
HISMetrics	Hersteller Initiative Software metrics
✓ IncludeMetrics	Metrics based on includes.
✓ IssueCounts	Metrics to count the number of issues by type.
✓ LinesOfCodeMetrics	Metrics based on lines of code.
✓ OOPMetrics	Metrics for object-oriented systems.
OtherMetrics	Various other metrics.
TokenMetrics	Metrics counting specific tokens.
Stylechecks	Checks regarding coding style, including Misra and other checks.
AutosarC++	Checks for Autosar guidelines for the use of the C++14 language in critical and safety-related systems.
C#	Checks for C# coding style.
Cert	SEI CERT Coding Standard Security Checks.
CodingStyle	Checks for coding style issues.
CompilerErrata	Checks whether your code is affected by known errata in native compilers.
CQM	Code-Quality-Management Stylechecks.
CWE	Checks for issues listed in the CWE - Common Weakness Enumeration
FaultDetection	Static checks for possible runtime errors.
Generic	Generic/Best practice checks.
Misra	Checks for Misra guidelines for the use of C/C++ in critical systems.
▨ Qt	Rulegroups for programs using Qt
SecureCoding	ISO-TS-17961 C Secure Coding

Figure 11. Rules that are included in this analysis, as seen in Axivion's Project Configuration tool (CC BY 4.0 Qt Group)

I chose rules based on relevance to this project. For stylechecks I chose only Qt stylechecks because, obviously, this is a Qt project. Qt stylechecks included Qt-Autosar, which may not be completely necessary at this point of development, as Autosar rules are generally meant to be used with critical systems such as automotive systems. But I still felt that Qt-Autosar is more relevant to this project at this point than for example, the AutosarC++. [95]

After the selection of rules, the analysis is executed by running Axivion's start_analysis.sh shell script. This will generate the results to the Axivion Suite Dashboard that is used for browsing through the results.

4.2. Expert Evaluation Procedure

The demo session was held on 25th of August in 2023 at the Qt Group's premises in Oulu, Finland, and remotely using a video conference call. Three participants attended physically and three participants attended via the call. After the meeting I shared the link to the questionnaire and consent form to the participants. After the experts had answered to the questionnaire the thematic analysis was started.

Demo description

This demo session was arranged because the software is still in such a state in which it cannot be shared conveniently to the experts for them to try it out themselves. In the demo session, I briefly showcased and explained the main features of the software, how they work, what features were left out for now, and gave some of my own ideas for further development. After the session, we discussed the missing features, and I received feedback of the demo. I also issued an online questionnaire (Appendix 1) with a research permit for informed consent and open questions about the demo and DTs.

4.2.1. Participants

All participants provided their background information in the first part of the questionnaire. The information is listed below in the Table 1:

Table 1. Backgrounds of the experts

Expert number	Years in current profession	Current work	education
Expert 1	10+	software development, applications, RTOS platforms, automotive software	B.Sc.
Expert 2	1	developing and maintaining the software stack of an embedded Linux system made by Qt Group	Master degree of Engineering
Expert 3	3	Product Marketing and Product Management	Master of Science in Economics
Expert 4	2	SW developer/architect for embedded, 3GPP and application protocols, RTOS, UI. Team lead and embedded SW development	M.Sc.

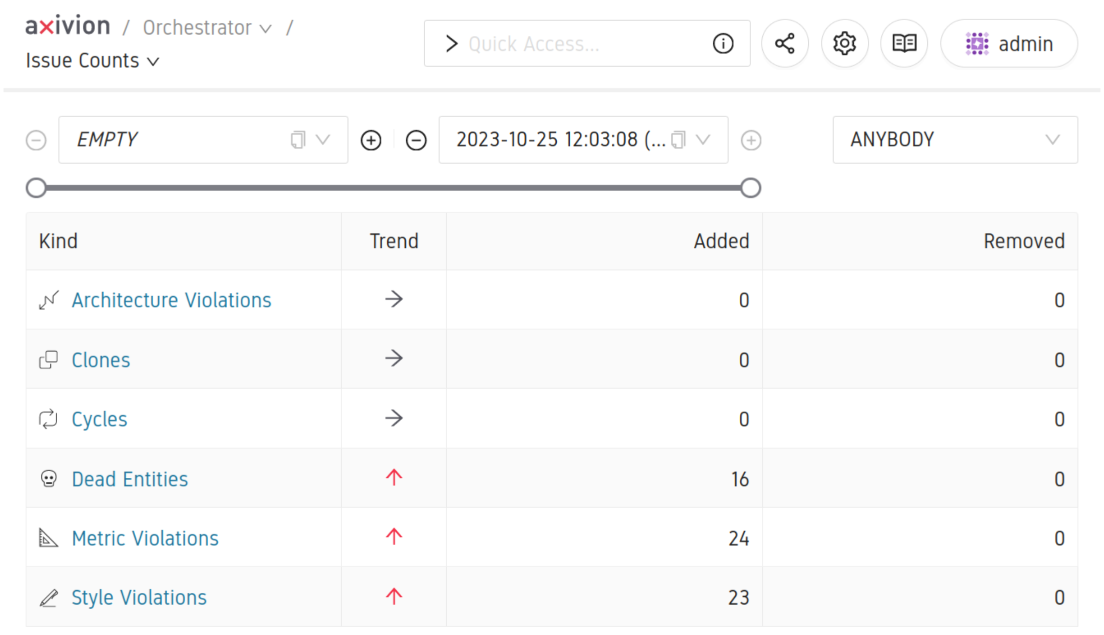
All the experts are male and working in Qt Group during the evaluation in August of 2023.

5. RESULTS

This chapter presents the results and analysis from both the quantitative technical evaluation and the qualitative expert evaluation [18]. The quantitive data was collected to assess the maturity of the demo, and these results will be discussed in the next subchapter.

5.1. Technical Evaluation

The results of the static code analysis by Axivion can be read from the Axivion Suite Dashboard. The included rules of the analysis can be seen from Figure 11 in the previous chapter. The results from the analysis can be seen from Figure 12 below.

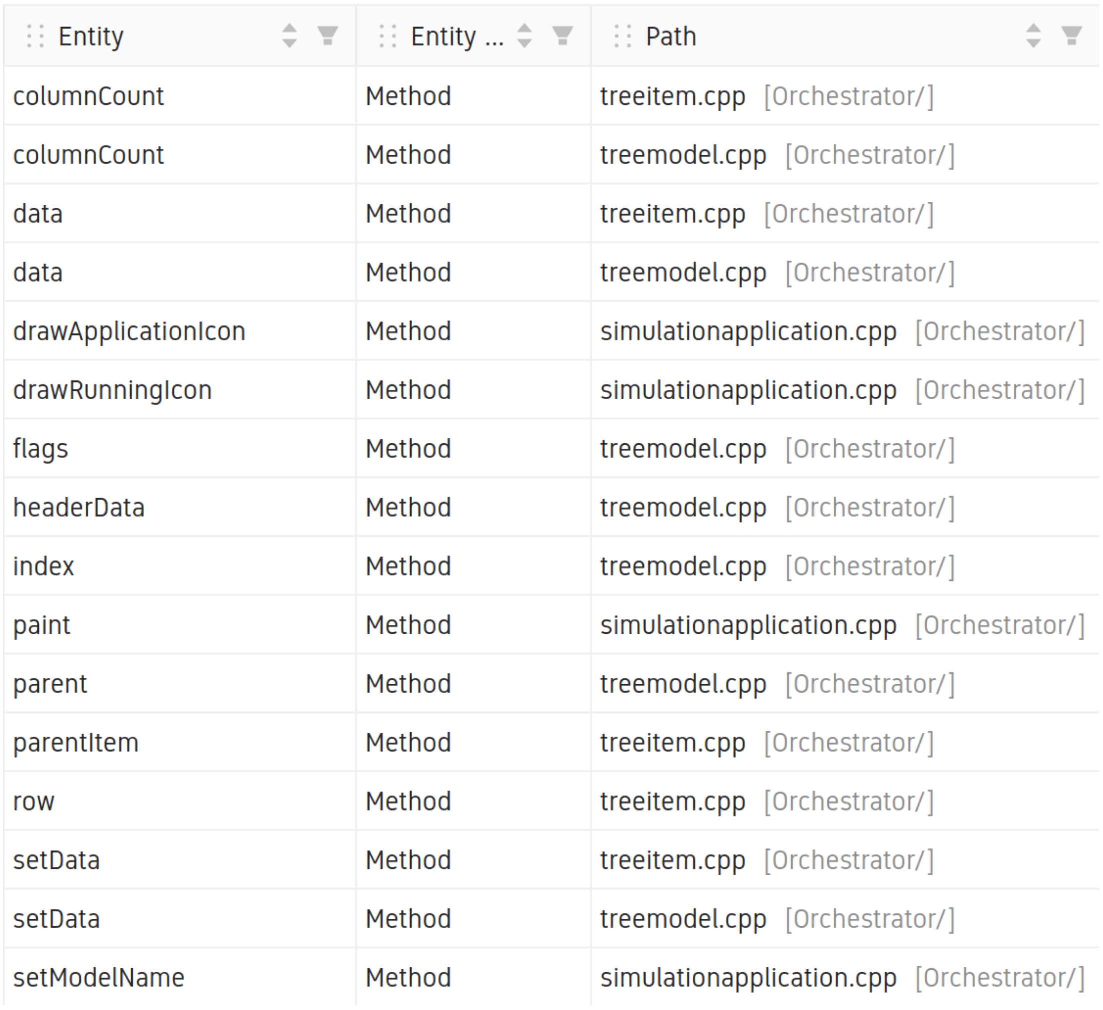


Kind	Trend	Added	Removed
Architecture Violations	→	0	0
Clones	→	0	0
Cycles	→	0	0
Dead Entities	↑	16	0
Metric Violations	↑	24	0
Style Violations	↑	23	0

Figure 12. The results of the static code analysis, as seen in Axivion Suite Dashboard (CC BY 4.0 Qt Group)

As can be seen, there are 63 issues found by Axivion in total. 16 dead code entities, 24 metric-based violations and 23 style violations. Axivion did not find any architecture violations, cloned code parts, "#include loops" or cycles in the call graph.

Dead code may have multiple definitions but Axivion considers dead code to be code that is never executed [96]. All the dead code entities found by Axivion can be found in the Figure 13 below. Many of the dead code entities seem to be getter functions.



Entity	Entity ...	Path
columnCount	Method	treeitem.cpp [Orchestrator/]
columnCount	Method	treemodel.cpp [Orchestrator/]
data	Method	treeitem.cpp [Orchestrator/]
data	Method	treemodel.cpp [Orchestrator/]
drawApplicationIcon	Method	simulationapplication.cpp [Orchestrator/]
drawRunningIcon	Method	simulationapplication.cpp [Orchestrator/]
flags	Method	treemodel.cpp [Orchestrator/]
headerData	Method	treemodel.cpp [Orchestrator/]
index	Method	treemodel.cpp [Orchestrator/]
paint	Method	simulationapplication.cpp [Orchestrator/]
parent	Method	treemodel.cpp [Orchestrator/]
parentItem	Method	treeitem.cpp [Orchestrator/]
row	Method	treeitem.cpp [Orchestrator/]
setData	Method	treeitem.cpp [Orchestrator/]
setData	Method	treemodel.cpp [Orchestrator/]
setModelName	Method	simulationapplication.cpp [Orchestrator/]

Figure 13. All 16 dead code entities, as seen in Axivion Suite Dashboard (CC BY 4.0 Qt Group)

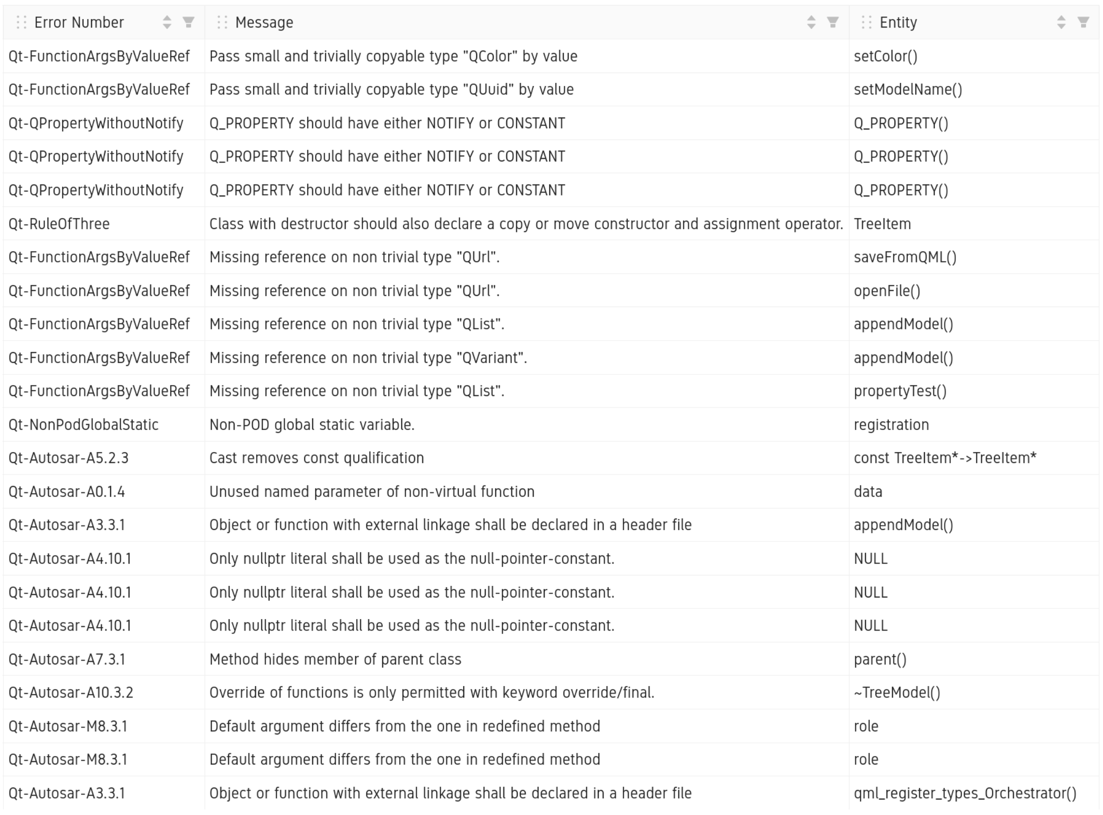
Style violations contain multiple different style violation types and there are two severity levels for these style violations, warning and required. There are 12 violations marked with warning and 11 marked with required severity. All the violations found by Qt-Autosar are required level and the violations found by Qt-Clazy are warnings. Qt-Generic's rules did not find any violations.

The rule Qt-FunctionArgsByValueRef of Qt-Clazy says that it warns when the value should be passed by value instead of by reference and vice-versa. There are three violations of rule Qt-QPropertyWithoutNotify of Qt-Clazy, which says that every Q_PROPERTY should have either a NOTIFY signal or a CONSTANT attribute defined.

There is a violation of a rule called Qt-RuleOfThree, which is triggered from a class TreeItem. This rule implements the rule of three which means that if a class defines a destructor, copy constructor or copy assignment operator, it should define them all [97, 98].

Qt-Autosar's rules detected multiple violations. For example, Qt-Autosar-A4.10.1 rule found three violations about using NULL instead of nullptr literal as a null-pointer-constant. Qt-Autosar-M8.3.1 rule found two violations about the difference between the default parameters of the virtual function and its overriding function. Qt-Autosar-A0.1.4 found a violation about an unused parameter of a non-virtual function. Qt-Autosar-A10.3.2 found a violation about not using the override keyword while overriding a function. The violation points to the destructor of the class TreeModel.

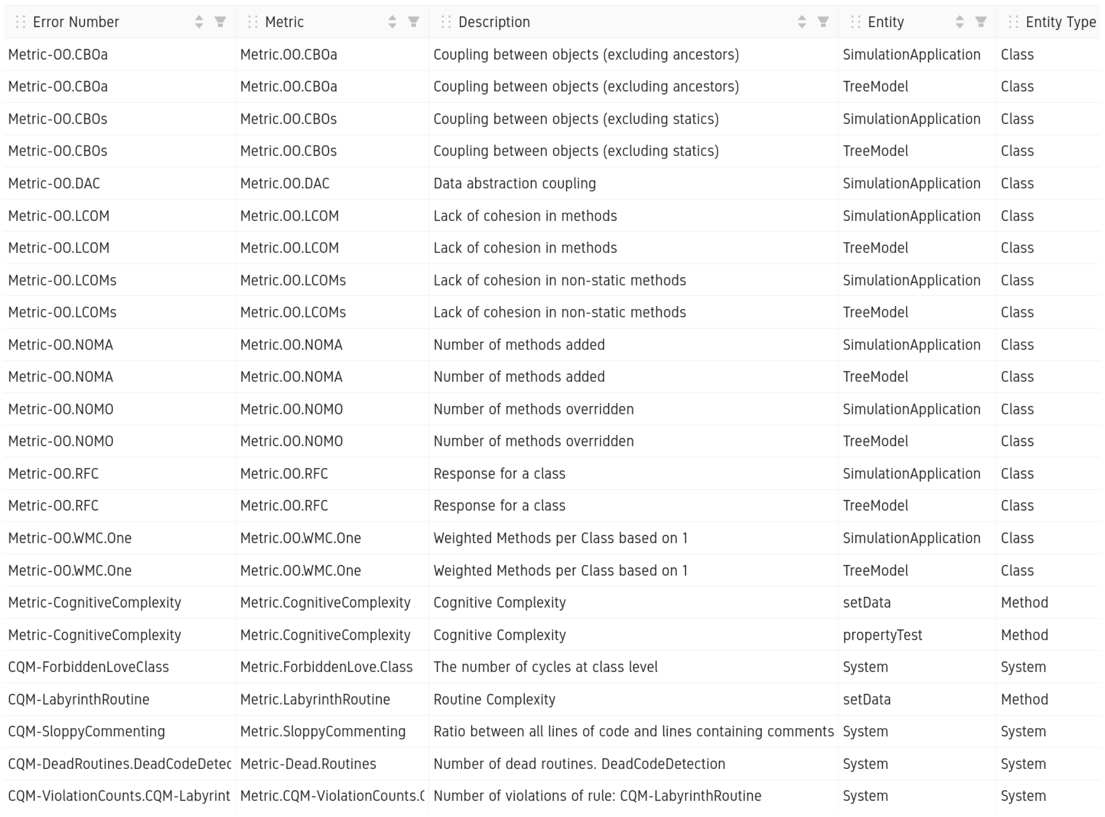
All style violations can be seen in the Figure 14 below.



Error Number	Message	Entity
Qt-FunctionArgsByValueRef	Pass small and trivially copyable type "QColor" by value	setColor()
Qt-FunctionArgsByValueRef	Pass small and trivially copyable type "QUuid" by value	setModelName()
Qt-QPropertyWithoutNotify	Q_PROPERTY should have either NOTIFY or CONSTANT	Q_PROPERTY()
Qt-QPropertyWithoutNotify	Q_PROPERTY should have either NOTIFY or CONSTANT	Q_PROPERTY()
Qt-QPropertyWithoutNotify	Q_PROPERTY should have either NOTIFY or CONSTANT	Q_PROPERTY()
Qt-RuleOfThree	Class with destructor should also declare a copy or move constructor and assignment operator.	TreeItem
Qt-FunctionArgsByValueRef	Missing reference on non trivial type "QUrl".	saveFromQML()
Qt-FunctionArgsByValueRef	Missing reference on non trivial type "QUrl".	openFile()
Qt-FunctionArgsByValueRef	Missing reference on non trivial type "QList".	appendModel()
Qt-FunctionArgsByValueRef	Missing reference on non trivial type "QVariant".	appendModel()
Qt-FunctionArgsByValueRef	Missing reference on non trivial type "QList".	propertyTest()
Qt-NonPodGlobalStatic	Non-POD global static variable.	registration
Qt-Autosar-A5.2.3	Cast removes const qualification	const TreeItem*->TreeItem*
Qt-Autosar-A0.1.4	Unused named parameter of non-virtual function	data
Qt-Autosar-A3.3.1	Object or function with external linkage shall be declared in a header file	appendModel()
Qt-Autosar-A4.10.1	Only nullptr literal shall be used as the null-pointer-constant.	NULL
Qt-Autosar-A4.10.1	Only nullptr literal shall be used as the null-pointer-constant.	NULL
Qt-Autosar-A4.10.1	Only nullptr literal shall be used as the null-pointer-constant.	NULL
Qt-Autosar-A7.3.1	Method hides member of parent class	parent()
Qt-Autosar-A10.3.2	Override of functions is only permitted with keyword override/final.	~TreeModel()
Qt-Autosar-M8.3.1	Default argument differs from the one in redefined method	role
Qt-Autosar-M8.3.1	Default argument differs from the one in redefined method	role
Qt-Autosar-A3.3.1	Object or function with external linkage shall be declared in a header file	qml_register_types_Orchestrator()

Figure 14. All the style violations, as seen in Axivion Suite Dashboard (CC BY 4.0 Qt Group)

Most of the metric-based violations are related to OOPMetrics that contain metrics for object oriented systems. A violation regarding cognitive complexity metric was found in two functions, setData and propertyTest, which are both found from the same file. There are also a violation about routine complexity from the function setData. All the metric-based violations can be found in Figure 15 below.



Error Number	Metric	Description	Entity	Entity Type
Metric-OO.CBOa	Metric.OO.CBOa	Coupling between objects (excluding ancestors)	SimulationApplication	Class
Metric-OO.CBOa	Metric.OO.CBOa	Coupling between objects (excluding ancestors)	TreeModel	Class
Metric-OO.CBOs	Metric.OO.CBOs	Coupling between objects (excluding statics)	SimulationApplication	Class
Metric-OO.CBOs	Metric.OO.CBOs	Coupling between objects (excluding statics)	TreeModel	Class
Metric-OO.DAC	Metric.OO.DAC	Data abstraction coupling	SimulationApplication	Class
Metric-OO.LCOM	Metric.OO.LCOM	Lack of cohesion in methods	SimulationApplication	Class
Metric-OO.LCOM	Metric.OO.LCOM	Lack of cohesion in methods	TreeModel	Class
Metric-OO.LCOMs	Metric.OO.LCOMs	Lack of cohesion in non-static methods	SimulationApplication	Class
Metric-OO.LCOMs	Metric.OO.LCOMs	Lack of cohesion in non-static methods	TreeModel	Class
Metric-OO.NOMA	Metric.OO.NOMA	Number of methods added	SimulationApplication	Class
Metric-OO.NOMA	Metric.OO.NOMA	Number of methods added	TreeModel	Class
Metric-OO.NOMO	Metric.OO.NOMO	Number of methods overridden	SimulationApplication	Class
Metric-OO.NOMO	Metric.OO.NOMO	Number of methods overridden	TreeModel	Class
Metric-OO.RFC	Metric.OO.RFC	Response for a class	SimulationApplication	Class
Metric-OO.RFC	Metric.OO.RFC	Response for a class	TreeModel	Class
Metric-OO.WMC.One	Metric.OO.WMC.One	Weighted Methods per Class based on 1	SimulationApplication	Class
Metric-OO.WMC.One	Metric.OO.WMC.One	Weighted Methods per Class based on 1	TreeModel	Class
Metric-CognitiveComplexity	Metric.CognitiveComplexity	Cognitive Complexity	setData	Method
Metric-CognitiveComplexity	Metric.CognitiveComplexity	Cognitive Complexity	propertyTest	Method
CQM-ForbiddenLoveClass	Metric.ForbiddenLove.Class	The number of cycles at class level	System	System
CQM-LabyrinthRoutine	Metric.LabyrinthRoutine	Routine Complexity	setData	Method
CQM-SloppyCommenting	Metric.SloppyCommenting	Ratio between all lines of code and lines containing comments	System	System
CQM-DeadRoutines.DeadCodeDetec	Metric-Dead.Routines	Number of dead routines. DeadCodeDetection	System	System
CQM-ViolationCounts.CQM-Labyrint	Metric.CQM-ViolationCounts.	Number of violations of rule: CQM-LabyrinthRoutine	System	System

Figure 15. All the metric-based violations, as seen in Axivion Suite Dashboard (CC BY 4.0 Qt Group)

5.2. Expert Evaluation: Qualitative Results

A questionnaire was issued after the demo session. The answers to the questionnaire (Appendix 1) provide the qualitative data for this research. A thematic analysis was done to this data.

The sample size was rather small since only four experts took part in answering the questionnaire. The answers contained 953 words in total, which forms the whole data for this analysis. The results of the expert evaluation questionnaire are analyzed using thematic analysis [19]. Every observation was assigned a code. The codes were then categorized, in other words, assigned to a theme group, represented by a common theme that connects the observations together. The themes and the codes are listed in the coding schema in Appendix 2. The number in brackets after a theme's or code's name indicates the number of observations belonging in that theme or code according to the researcher's (i.e. my) interpretation.

General feedback (16) is a theme that consists mostly of observations that contain feedback about the demo session and the Orchestrator in a broad way. Some general feedback from experts 1 and 2:

> *"Good impressive demo, especially integrating it to existing Qt applications via this SimulationItem type is a intuitive feature. "*
> *- Expert 1*

> *"An impresive demo, though the UI needs some polishing. The functions are implemented and demostrated already. The highlight is the controller software(the "orchestrator") can fully control and be aware of the programs it creates."*
> *- Expert 2*

Variability and abundance of solutions (8) consists of development ideas for digital twin interfaces in general. The experts were asked if they think that there should be more solutions for interfacing digital twins.

> *"Yes, but they need to be interoperable or have the ability to connect to devices from multiple vendors. That is where a lot of the complexity comes in and where an interfacing technology plays the biggest role."*
> *- Expert 3*

> *"Yes, every SW development framework, like Qt FW [short for framework], should provide support for digital twin use cases. Current solutions from commercial vendors provide generic solutions that don't integrate well with specific SW development frameworks like Qt FW. The level of integration is just too high to support well the used SW development FW."*
> *- Expert 4*

Potential and development targets (29) consists of the specific development targets that the expert sees as a desired features for the system. As can be seen from the Appendix 2, this theme has a lot of codes and observations. A few ideas from experts 3 and 4:

> *"5/6G is definitely the biggest. A lot of factories are going to remote monitoring systems and this is a must. Another is edge more than cloud as they need these machines to be able to react based on sensor data, etc. and even the slightest latency or risk of loss of communication to a cloud is not acceptable. Augmented reality can also be very useful for validation of the equipment."*
> *- Expert 3*

> *"Possibility to launch Qt applications in an emulated Android device. Possibly also to be able to control also physical devices (turn ON/OFF, configure). Integration to simulated Qt APIs to monitor and control virtual devices (e.g. BT)."*
> *- Expert 4*

Use cases (19) consists of experts' ideas on where they could see the Orchestrator being used as of right now or after further development, or use cases of DTs in general. The most frequently occurring idea was related to demos. The experts think that they could see the Orchestrator being used for building Qt demos and demoing Qt technology to customers.

> *"Simulating, controlling and monitoring multiple digital twin and real-world applications e.g. in industrial automation applications."*
> *- Expert 1*

"As a part of building the demos and to connect all of the different devices/ HMIs built with Qt"
- Expert 3

5.2.1. Agreement Testing

The results of the thematic analysis are already in the Appendix 2 but with all qualitative data analyses there are interpretability involved. To assess this interpretability, an agreement testing process is done. In this case, an agreement testing between two observers with Cohen's Kappa [94].

Interpretibility of the themes and codes was evaluated by agreement testing, which in this case, involved two observers. The two interpretations of the material were compared against each other by calculating Cohen's Kappa from the observations, which reveals the agreement between the observers, including the agreement on the themes themselves and the agreement on the individual codes within the themes. In this case, the observations are divided in four categories which means that five calculations were needed. One calculation for measuring the agreement on which observation belongs to which theme and one calculation for each theme, for the agreement about, which observation belongs to which code within the theme.

The calculations were done using Cohen's Kappa calculator by GraphPad [99]. The results of the calculated Kappa coefficients were then compared against the scale presented in the table 2 below [100].

Themes

Totally, there were 72 observations that were assigned in a theme group. The number of observed agreements with the second observer is 64 which is 88.89% of the observations. The number of agreements expected by chance is 21.3 which is 29.55%. The weighted Kappa is 0.882 which refers to almost perfect agreement (Table 2).

General feedback

Totally, there were 16 observations of general feedback in the calculation. The number of observed agreements with the second observer is 8 which is 50.00% of the observations. The number of agreements expected by chance is 3.8 which is 23.83%. The weighted Kappa is 0.229 which refers to a fair agreement (Table 2).

Variability and abdundance of solutions

In this theme there were 10 observations taken into account in the calculation. The number of observed agreements with the second observer is 7 which is 70.00% of the observations. The number of agreements expected by chance is 2.2 which is 22.00%. The weighted Kappa is 0.529 which refers to a moderate agreement (Table 2).

Potential and development targets

In this calculation, 34 observations were involved. The number of observed agreements with the second observer is 29 which is 85.29% of the observations. The number of agreements expected by chance is 7.6 which is 22.32%. The weighted Kappa is 0.822 which refers to almost perfect agreement (Table 2).

Use cases

In total, there were 20 observations of use cases in the calculation. The number of observed agreements with the second observer is 15 which is 75.00% of the observations. The number of agreements expected by chance is 3.7 which is 18.50%. The weighted Kappa is 0.632 which refers to a substantial agreement (Table 2).

Table 2. Verbal interpretation of the Kappa scale [100]

Value	**Agreement level**
Kappa < 0	No agreement
0.00 < Kappa < 0.20	Slight agreement
0.21 < Kappa < 0.40	Fair agreement
0.41 < Kappa < 0.60	Moderate agreement
0.61 < Kappa < 0.80	Substantial agreement
0.81 < Kappa < 1.00	Almost perfect agreement

6. DISCUSSION

This project's main focus was the development of the Qt-based Digital Twin Orchestrator GUI software, which enables the users to begin creating their own DT projects. An expert evaluation [18] was executed based on some questions about the software as well as DTs in general. The results were confirmed by agreement testing with Cohen´s Kappa analysis [94]. The code base was also analyzed statically using Axivion Suite to provide feedback about the code.

6.1. Insights from the Analyses

Some interesting insights emerged both from the results of the expert evaluation and the results of the technical evaluation. Some of these are worth discussing.

6.1.1. Domain Experts´ Views on DTs

The experts were asked about the definition of a DT. From the answers, a general picture about the definition was formed. According to the experts, a DT is a digital copy of a physical system. It is used for product development, prototyping, testing and maintaining a product. In product development, it can be used to develop the product before hardware is available. I think this definition relates strongly to the practical industrial knowledge that the experts are most likely familiar with.

6.1.2. Experts' Views on the Future of DTs

When asked of further development targets of DTs in general, all the experts mentioned AR, and only one expert mentioned VR and even that was mentioned in conjunction with AR. AR might be more prominent than VR in the industrial context or in some context that all these experts are familiar with. This observation is also suported by literature [101, 102, 103].

Three out of four experts mentioned customer demos as a use case for the Digital Twin Orchestrator. I think the reason is that Qt Group makes lots of demos for Qt technology. Many of these demos are then shown to customers or potential customers in, for example, during exhibits. I think this must be a familiar use case for all the experts.

6.1.3. Insights Based on Technical Analysis

There are many tools that can analyze C++ code statically, such as SonarQube, CodeScene and Coverity [104, 105, 106]. Axivion was chosen as the tool for the static code analysis because it was bought by Qt Group in 2022, and it has the analysis rules for Qt projects, such as Qt stylechecks and Qt framework violation checks, which I felt were relevant to this project as it is a Qt project.

The results of this static code analysis are not very significant in terms of making further analysis, but I think they provide good feedback to help the programmer improve the code, and I think this is the main purpose of static code analysis. There are lots of rules which can be included in the analysis, and of course, the selection of rules heavily affects the relevancy and the amount of results. Using more rules would most likely lead to detecting more violations. I tried to include those which I thought were the most relevant and would provide the most relevant feedback. I think the relevancy of the rules really depends on the project itself. Every project has a different emphasis on what kinds of violations are the most crucial to know for the project, and what are not so important. I think this static code analysis was a really good opportunity to get feedback on the quality of the code, and I could clearly see what things I still need to pay more attention to when I am programming Qt programs and C++ in general. For example, even with these limited rules that were used in the analysis, Axivion still found multiple violations for the function setData of the class TreeModel. I think this is a pretty clear indication that the function should be improved. There were also many violations about passing a variable to function by value or by reference when it should have been done the opposite way. This is a clear sign of something I need to pay more attention to.

However, the static code analysis may not be perfect. I got a few violations stating that I have used differing default parameters of overriding virtual functions in the C++ code, although it was checked multiple times that the declarations of these overriding functions are identical to the virtual function declarations in the Qt documentation. One reason may be that the Qt-Autosar uses some older version of Qt where some of these virtual function declarations may use different default parameters to what they are using currently, and as a result, Axivion finds a violation. But this was not checked, so this is just a speculation.

6.2. Further Development

Both the implementation of the Digital Twin Orchestrator and the analyses can be improved by further development. Lots of development ideas emerged for the Orchestrator from the experts' answers to the questionnaire, discussing the project with the experts and while doing the background research.

6.2.1. Further Development of the Digital Twin Orchestrator

Some features of the Orchestrator were left deliberately for further development, such as the Bluetooth tunneling feature. This would allow the users to connect their own smart devices to the Orchestrator via Bluetooth. Then, they would be able to control, for example, the robot arm application with their physical smart devices.

At least one expert said that, at some point, the Orchestrator should be able to integrate physical objects into the DT project. This would be, of course, a very important feature because a DT needs the physical object as it is one of the three main parts of a DT. One idea was that the user should be able to create a gateway item to the simulation space that could be used to communicate with the physical object. There was initially the purpose to include this feature, but it was left out to limit the workload of the project.

Currently, the software does not take part in creating the virtual model of the twin. A user has to provide their own application, which needs to include and instantiate the custom SimulationItem plugin. From the beginning, the idea was to keep the Orchestrator as generic and flexible as possible, meaning that the user could have a lot of choices about what to do with the software. The Orchestrator only takes part in one of the three parts of a DT, the connections between the physical and virtual objects. Depending on which direction the development of this software goes, it may be necessary to add more features regarding the virtual objects.

One of the main further development ideas that I got after the demo and that was also discussed among the experts was the choice of the communication protocol between the host and client nodes. In this project within the SimulationItem plugin, Qt Remote Objects were used for the data transfer, which is a Qt-specific communication method. The industry standards tend to lean more towards other more common and widely used protocols in IIoT applications, such as MQTT or OPC UA [7, 107, 17]. Using one of these more common protocols would mean that the SimulationItem plugin in its current form would have to be discarded. The hosts would need to send data based on these protocols, and the client would need to implement a new way of receiving the values in real-time. Also, the property bindings that have to be done in the host application for the SimulationItem to share the updating data are a Qt-specific mechanism. Using the QtRO might tie the customers more tightly to the Qt environment.

Another option would be for the hosts and client to connect using WebSockets, where the host would send serialized dataframes to the client using the WebSocket application programming interface (API). The client would unpack the dataframes using the same API and then it could do whatever it wanted with the data. In this case, it would display it in the property view of the Digital Twin Orchestrator [108].

The Digital Twin Orchestrator aims to be a generic DT platform where there would be lots of options for the user to create their own DT environments by connecting physical objects to already-made virtual replicas. The purpose was not to focus on creating the virtual model of a DT through 3D modeling. It was left to the user's responsibility to provide their own 3D environment and focus on integrating this environment with the platform. I think the end goal for the Digital Twin Orchestrator is quite similar to all the DT platforms that were presented. Maybe Nvidia's Omniverse was the most different of the platforms by focusing more on the real-time 3D and physics simulation, multi-user real-time collaboration, as well as the metaverse concept, which is an even wider concept than a DT. The Azure Digital Twins may be the most generic platform by focusing mostly on the structure and relationships of a DT but Microsoft provides a large amount of services in the Azure cloud platform that can be easily integrated to the Azure Digital Twins, such as the 3D Scenes Studio. But because of the wide scope of these platforms, similar DT goals could most likely be achieved by choosing any of the researched platforms. And, of course, it remains to be seen in which direction the development of the Digital Twin Orchestrator will be heading in the future.

6.2.2. Further Development of the Research

Quantitative data

In the future, I think it would be a good idea to add some more generic C++ set of rules as well, for example, the C++ Core Guidelines, to the static code analysis [109]. Depending on which direction the program development goes, it may be a good idea to add some set of rules designed for critical or safety-related systems such as MisraC++ or AutosarC++, which both have support in Axivion Suite [110, 95].

Qualitative data

The coding schema of the thematic analysis was improved after the first iteration of the analysis, together with the other observer. New codes were added and some old codes were removed. The important Other code was added under each theme. Its purpose is to hold many observations, which were not common enough for them to have their own code. For example, an observation was assigned to the Other code if it was mentioned in the data only once. We also decided on a procedure that if we do not agree on the theme of the current observation, the observations get assigned to the Other code during that calculation.

Other than this one improvement discussion before the second iteration, the analyses were done independently by both participants and the results of the second iteration were later confirmed by calculating the Cohen's Kappa coefficients representing the agreement levels.

The coding schema of the thematic analysis could, of course, be improved even further, and the analysis could be improved with more iterations. This would result into higher level conceptualization of the results.

6.3. Limitations

There were both technical and research-related limitations related to this project. It is worth noting these. Some initial plans for the development of the software had to be left out. There were lots of limitations to the research as well.

6.3.1. Research Limitations

Biases

Because I have been very closely related to the development process of this system, the thematic analysis results may be biased at least from my point of view. The second observer is also familiar with the thesis subject and the sample data, so this may have also affected some of the results. Agreement testing is often done with a participant who is not initially familiar with the data.

The fact that we developed and improved the themes and the coding schema together with the other observer might explain why the agreements are on a high level, especially on the calculation of the themes themselves, where the agreement was almost perfect according to the scale in table 2. In addition the codes are still directly deductable from the material, which would have changed if the analysis would have been continued.

Sample size

The sample size of this expert evaluation questionnaire was very small. Only four experts took part in this questionnaire and therefore, the results can not be applied to average users but only to this very small amount of people with this specific knowledge, which, of course, is the goal of an expert analysis.

Agreement testing

When calculating Cohen's Kappa for individual themes in the agreement testing, we decided on a procedure that if there is not an agreement on the current observation's theme, the observation is included in both themes' calculations. In this case, the observation would be placed in the Other code if one of the observers did not agree that the observation should be in the current theme in the first place. This also explains why the total number of observations in a specific theme does not match between the theme calculation and the individual calculations of a specific theme. If the other observer thinks that an observation belongs to a specific theme while I do not, the observation still gets assigned to the Other code by myself. So, in the end, it may look like we have an agreement on the observation's theme and code, but in reality, we do not agree even on the theme. This may also explain why most of the agreement Kappa coefficients are as high as they are. This is important to note because, without this decided procedure, the Kappa coefficients would be lower.

Because the coding schema was not improved further via further development iterations, the codes were still clearly connected to the text. This may have been one more reason why the agreement Kappa coefficients were so high.

The lowest agreement level was on general feedback with a weighted Kappa of only 0.229, meaning fair agreement, whereas the highest agreement level was on the themes themselves with a weighted Kappa of 0.882, meaning almost perfect agreement. Potential and development target reached almost as high with a weighted Kappa of 0.822, also meaning almost perfect agreement. The agreement on the themes may have been as high because there were only four themes included. The themes may not have been too similar to each other, so the observations were most likely easy to place within the themes.

The agreement on potential and development targets may have been so high because the observations were thought about with the other observer for other reasons even before the agreement testing.

6.3.2. Technical Limitations

This project was not a full DT project. In order to be a real DT, the project should have included the physical part of the twin as well. This project focused only on connecting the virtual part of the DT with the Digital Twin Orchestrator. The ability to connect and control a physical robot arm was left for further development. If the project had included a full DT with the physical part, the results of the questionnaire could have been different. The experts could have come up with different future development ideas or have different views on the potential of the project. On the other hand, now the demo was not too spesific or restricting, which allowed more creative contemplation on the use other technologies for interfacing with DTs.

7. SUMMARY

This work presents the concept of a Digital Twin (DT), many related technologies, and a brief history of DTs. This work presents few examples of how DTs are currently used in the industry and what kind of benefits companies claim to have gained by using DTs. Some of the most popular DT platforms are also presented.

In this study a GUI software (Digital Twin Orchestrator) was created for managing DT environments. Quantitative data was gathered from the software by analyzing it statically via Axivion Suite, finding some violations from the presented code base.

The functionality of the software was demonstrated to domain experts, and after the presentation, the experts had to answer a questionnaire consisting of questions about DTs in general and about the Digital Twin Orchestrator. These answers, were analyzed thematically by assigning every observation a theme and a code by the researcher. The interpretability of these results was then assessed by agreement testing between two observers with Cohen's Kappa, calculating Cohen's Kappa coefficients representing the agreement level between the two observers.

The experts agreed on the potential of similar DT interfaces for the industry. They expressed more interest in AR interfaces than VR interfaces or other Industry 4.0 technologies. They saw great potential of these technologies in the industry and for other spesific contexts such as PR. This information may be useful to future developers and researchers of DTs.

8. REFERENCES

[1] Leng J., Wang D., Shen W., Li X., Liu Q. & Chen X. (2021) Digital twins-based smart manufacturing system design in industry 4.0: A review. Journal of Manufacturing Systems 60, pp. 119–137. URL: https://www.sciencedirect.com/science/article/pii/S0278612521001151.

[2] Tao F., Xiao B., Qi Q., Cheng J. & Ji P. (2022) Digital twin modeling. Journal of Manufacturing Systems 64, pp. 372–389. URL: https://www.sciencedirect.com/science/article/pii/S0278612522001108.

[3] Menon D., Anand B. & Chowdhary C.L. (2023) Digital twin: Exploring the intersection of virtual and physical worlds. IEEE Access 11, pp. 75152–75172.

[4] Tao F., Zhang M. & Nee A.Y.C. (2019) Digital twin driven smart manufacturing. Academic Press.

[5] Qi Q., Tao F., Hu T., Anwer N., Liu A., Wei Y., Wang L. & Nee A. (2021) Enabling technologies and tools for digital twin. Journal of Manufacturing Systems 58, pp. 3–21. URL: https://www.sciencedirect.com/science/article/pii/S027861251930086X, digital Twin towards Smart Manufacturing and Industry 4.0.

[6] Phanden R., Sharma P. & Dubey A. (2020) A review on simulation in digital twin for aerospace, manufacturing and robotics. Materials Today: Proceedings 38.

[7] Kritzinger W., Karner M., Traar G., Henjes J. & Sihn W. (2018) Digital twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine 51, pp. 1016–1022. URL: https://www.sciencedirect.com/science/article/pii/S2405896318316021, 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.

[8] Fuller A., Fan Z., Day C. & Barlow C. (2020) Digital twin: Enabling technologies, challenges and open research. IEEE Access 8, pp. 108952–108971.

[9] Pires F., Cachada A., Barbosa J., Moreira A.P. & Leitão P. (2019) Digital twin in industry 4.0: Technologies, applications and challenges. In: 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), vol. 1, vol. 1, pp. 721–726.

[10] V P., P L., Jain N.J., L P N., H N. & K B. (2021) Digital twin technology: A bird eye view. In: 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1069–1075.

[11] Metso Outotec launches Geminex™, a digital twin for efficient management of variability in mining and metallurgical operations. https://www.metso.com/corporate/media/news/2022/3/metso-outotec-launches-geminex-a-digital-twin-for-efficient-management-of-variability-in-mining-and-metallurgical-operations/. Accessed: 2023-07-18.

[12] Geminex. https://www.metso.com/mining/solutions/geminex/. Accessed: 2023-07-18.

[13] Vehicle Digital Twin: when physical and digital models unite. https://www.renaultgroup.com/en/news-on-air/news/vehicle-digital-twin-when-physical-and-digital-models-unite/. Accessed: 2023-07-20.

[14] Case Framery Digital Twin Prototyping, The Future of World Class Product Design. https://varjo.com/blog/framery-iterative-mixed-reality-digital-twin-prototyping/. Accessed: 2023-09-26.

[15] Ibrahim M., Rjabtšikov V. & Gilbert R. (2023) Overview of digital twin platforms for ev applications. Sensors 23. URL: https://www.mdpi.com/1424-8220/23/3/1414.

[16] Liu C., Vengayil H., Lu Y. & Xu X. (2019) A cyber-physical machine tools platform using opc ua and mtconnect. Journal of Manufacturing Systems 51, pp. 61–74. URL: https://www.sciencedirect.com/science/article/pii/S0278612518301110.

[17] Unified Architecture. https://opcfoundation.org/about/opc-technologies/opc-ua/. Accessed: 2023-09-4.

[18] Väänänen-Vainio-Mattila K. & Wäljas M. (2009) Developing an expert evaluation method for user experience of cross-platform web services. In: Proceedings of the 13th International MindTrek Conference: Everyday Life in the Ubiquitous Era, MindTrek '09, Association for Computing Machinery, New York, NY, USA, p. 162–169. URL: https://doi-org.pc124152.oulu.fi:9443/10.1145/1621841.1621871.

[19] Braun V. & Clarke V. (2006) Using thematic analysis in psychology. Qualitative research in psychology 3, p. 77–101.

[20] Axivion. https://www.axivion.com/en/. Accessed: 2023-10-19.

[21] Louridas P. (2006) Static code analysis. IEEE Software 23, pp. 58–61.

[22] Hermann M., Pentek T. & Otto B. (2016) Design principles for industrie 4.0 scenarios. In: 2016 49th Hawaii International Conference on System Sciences (HICSS), pp. 3928–3937.

[23] Bassi L. (2017) Industry 4.0: Hope, hype or revolution? In: 2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI), pp. 1–6.

[24] Tao F., Qi Q., Wang L. & Nee A. (2019) Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. Engineering 5, pp. 653–661. URL: `https://www.sciencedirect.com/science/article/pii/S209580991830612X`.

[25] Qin H., Wang H., Zhang Y. & Lin L. (2021) Constructing digital twin for smart manufacturing. In: 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 638–642.

[26] Ichhuda K., Shah K. & Kumar A. (2021) Cloud-based smart manufacturing: Implementation in food industry. In: 2021 2nd International Conference for Emerging Technology (INCET), pp. 1–5.

[27] Drath R. & Horch A. (2014) Industrie 4.0: Hit or hype? [industry forum]. IEEE Industrial Electronics Magazine 8, pp. 56–58.

[28] Al-Sarawi S., Anbar M., Alieyan K. & Alzubaidi M. (2017) Internet of things (iot) communication protocols: Review. In: 2017 8th International Conference on Information Technology (ICIT), pp. 685–690.

[29] Kopetz H. & Steiner W. (2022) Internet of Things, Springer International Publishing, Cham. pp. 325–341. URL: `https://doi.org/10.1007/978-3-031-11992-7_13`.

[30] Xu L.D., He W. & Li S. (2014) Internet of things in industries: A survey. IEEE Transactions on Industrial Informatics 10, pp. 2233–2243.

[31] Hussain S. & Chaudhry S.A. (2019) Comments on "biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment". IEEE Internet of Things Journal 6, pp. 10936–10940.

[32] Lasi H. Fettke P. K.H. (2014) Industry 4.0 .

[33] Ray P.P. (2018) An introduction to dew computing: Definition, concept and implications. IEEE Access 6, pp. 723–737.

[34] M A., Dinkar A., Mouli S.C., B S. & Deshpande A.A. (2021) Comparison of containerization and virtualization in cloud architectures. In: 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), pp. 1–5.

[35] Cloud Scalability. `https://www.vmware.com/topics/glossary/content/cloud-scalability.html`. Accessed: 2023-03-13.

[36] Yen C.T., Liu Y.C., Lin C.C., Kao C.C., Wang W.B. & Hsu Y.R. (2014) Advanced manufacturing solution to industry 4.0 trend through sensing network and cloud computing technologies. In: 2014 IEEE International Conference on Automation Science and Engineering (CASE), pp. 1150–1152.

[37] Types of Cloud Computing. `https://aws.amazon.com/types-of-cloud-computing/`. Accessed: 2023-03-14.

[38] Pan J. & McElhannon J. (2018) Future edge cloud and edge computing for internet of things applications. IEEE Internet of Things Journal 5, pp. 439–449.

[39] Bajic B., Cosic I., Katalinic B., Moraca S., Lazarevic M. & Rikalovic A. (2019) Edge computing vs. cloud computing: Challenges and opportunities in industry 4.0. Annals of DAAAM & Proceedings 30.

[40] Tao F., Cheng J., Qi Q., Zhang M., Zhang H. & Sui F. (2018) Digital twin-driven product design, manufacturing and service with big data. The International Journal of Advanced Manufacturing Technology 94, pp. 3563–3576.

[41] Mateev M. (2020) Industry 4.0 and the digital twin for building industry. Industry 4.0 5, pp. 29–32.

[42] Yousefnezhad N., Malhi A., Kinnunen T., Huotari M. & Främling K. (2020) Product lifecycle information management with digital twin: A case study. In: 2020 IEEE 18th International Conference on Industrial Informatics (INDIN), vol. 1, vol. 1, pp. 321–326.

[43] Mohammadi N. & Taylor J.E. (2017) Smart city digital twins. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–5.

[44] Sutani, Ramadhan F., Hidayat F., Hakim D.N., Amaliah U. & Kuntoro W.S. (2022) Study case of smart city: The usage of digital twin in transportation system. In: 2022 International Conference on Information Technology Systems and Innovation (ICITSI), pp. 113–117.

[45] Tao F., Zhang M., Liu Y. & Nee A. (2018) Digital twin driven prognostics and health management for complex equipment. CIRP Annals 67, pp. 169–172. URL: https://www.sciencedirect.com/science/article/pii/S0007850618300799.

[46] What is a digital twin? https://www.ibm.com/topics/what-is-a-digital-twin. Accessed: 2023-05-19.

[47] Bao L., Wang Q. & Jiang Y. (2021) Review of digital twin for intelligent transportation system. In: 2021 International Conference on Information Control, Electrical Engineering and Rail Transit (ICEERT), pp. 309–315.

[48] Hayat M. & Winkler H. (2022) Exploring the basic features and challenges of traditional product lifecycle management systems. In: 2022 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), pp. 0762–0766.

[49] Främling K., Holmström J., Ala-Risku T. & Kärkkäinen M. (2003) Product agents for handling information about physical objects. Report of Laboratory of information processing science series B, TKO-B 153.

[50] Saaksvuori A. & Immonen A. (2008) Product lifecycle management. Springer Science & Business Media.

[51] KeyShot. https://www.keyshot.com/. Accessed: 2023-09-26.

[52] KeyVR. https://www.keyshot.com/keyvr/. Accessed: 2023-09-26.

[53] Nath S.V., Van Schalkwyk P. & Isaacs D. (2021) Building industrial digital twins: Design, develop, and deploy digital twin solutions for real-world industries using Azure digital twins. Packt Publishing Ltd.

[54] GE Predix Platform. https://www.ge.com/digital/iiot-platform. Accessed: 2023-09-15.

[55] Siemens Insights Hub. https://plm.sw.siemens.com/en-US/insights-hub/. Accessed: 2023-11-14.

[56] Azure Digital Twins. https://azure.microsoft.com/en-us/products/digital-twins. Accessed: 2023-09-15.

[57] NVIDIA Omniverse. https://www.nvidia.com/en-us/omniverse/. Accessed: 2023-11-22.

[58] AWS IoT TwinMaker. https://aws.amazon.com/iot-twinmaker/. Accessed: 2023-11-22.

[59] Guo Y., Klink A., Bartolo P. & Guo W.G. (2023) Digital twins for electro-physical, chemical, and photonic processes. CIRP Annals 72, pp. 593–619. URL: https://www.sciencedirect.com/science/article/pii/S0007850623001361.

[60] Pfeiffer J., Lehner D., Wortmann A. & Wimmer M. (2022) Modeling capabilities of digital twin platforms-old wine in new bottles? J. Object Technol 21, p. 3.

[61] Issa R., S.Hamad M. & Abdel-Geliel M. (2023) Digital twin of wind turbine based on microsoft® azure iot platform. In: 2023 IEEE Conference on Power Electronics and Renewable Energy (CPERE), pp. 1–8.

[62] Meijers A. (2022) Hands-On Azure Digital Twins: A practical guide to building distributed IoT solutions. Packt Publishing Ltd.

[63] Digital Twins Definition Language (DTDL). https://azure.github.io/opendigitaltwins-dtdl/DTDL/v3/DTDL.v3.html. Accessed: 2023-11-07.

[64] Nölle C. & Kannisto P. (2023), Timeseries on iiot platforms: Requirements and survey for digital twins in process industry.

[65] Azure Digital Twins Explorer. https://learn.microsoft.com/en-us/azure/digital-twins/concepts-azure-digital-twins-explorer. Accessed: 2023-11-07.

[66] How Azure IoT enables business resilience. https://azure.microsoft.com/en-us/blog/how-azure-iot-enables-business-resilience/. Accessed: 2023-11-08.

[67] Insights Hub Basics. https://documentation.mindsphere.io/MindSphere/concepts/index.html. Accessed: 2023-11-16.

[68] MindConnect Library 3.1 for MindSphere 3.0. https://support.industry.siemens.com/cs/document/109755348/mindconnect-library-3-1-for-mindsphere-3-0-?dti=0&lc=en-US. Accessed: 2023-11-15.

[69] Insights Hub Connectivity. https://documentation.mindsphere.io/MindSphere/connectivity/overview.html. Accessed: 2023-11-21.

[70] Mourtzis D., Panopoulos N., Angelopoulos J., Wang B. & Wang L. (2022) Human centric platforms for personalized value creation in metaverse. Journal of Manufacturing Systems 65, pp. 653–659. URL: https://www.sciencedirect.com/science/article/pii/S0278612522001959.

[71] Liu L., Song X., Zhang C. & Tao D. (2022) Gan-mdf: An enabling method for multifidelity data fusion. IEEE Internet of Things Journal 9, pp. 13405–13415.

[72] Ericsson Builds Digital Twins for 5G Networks in NVIDIA Omniverse. https://blogs.nvidia.com/blog/ericsson-digital-twins-omniverse/. Accessed: 2023-11-27.

[73] Developer Guide Overview. https://docs.omniverse.nvidia.com/dev-guide/latest/index.html. Accessed: 2023-11-28.

[74] AWS IoT TwinMaker Documentation. https://docs.aws.amazon.com/iot-twinmaker/. Accessed: 2023-11-23.

[75] AWS IoT TwinMaker Features. https://aws.amazon.com/iot-twinmaker/features/. Accessed: 2023-11-23.

[76] AWS IoT SiteWise. https://aws.amazon.com/iot-sitewise/. Accessed: 2023-11-23.

[77] AWS IoT Device communication protocols. https://docs.aws.amazon.com/iot/latest/developerguide/protocols.html. Accessed: 2023-11-23.

[78] ESP32. https://www.espressif.com/en/products/socs/esp32. Accessed: 2023-05-26.

[79] Alvin A.I. robot for STEM/STEAM from Simua. https://simua.com/12345678-alvin-a-i-robot-for-stem-steam-from-simua. Accessed: 2023-05-5.

[80] C++ programming language. URL: https://cplusplus.com/, [C++17].

[81] Qt 6.5. https://doc.qt.io/qt-6/index.html. Version: 6.5.2.

[82] Qt Quick. https://doc.qt.io/qt-6/qtquick-index.html. Version: 6.5.2.

[83] Qt QML. https://doc.qt.io/qt-6/qtqml-index.html. Version: 6.5.2.

[84] CMake. https://cmake.org/. Version: 3.24.2.

[85] Build with CMake. https://doc.qt.io/qt-6/cmake-manual.html.

[86] Qt Creator Manual. https://doc.qt.io/qtcreator/.

[87] QAbstractItemModel Class. https://doc.qt.io/qt-6/qabstractitemmodel.html. Accessed: 2023-05-31.

[88] TreeView QML Type. https://doc.qt.io/qt-6/qml-qtquick-treeview.html. Accessed: 2023-05-31.

[89] QProcess Class. https://doc.qt.io/qt-6/qprocess.html. Accessed: 2023-05-31.

[90] Signals & Slots. https://doc.qt.io/qt-6/signalsandslots.html. Accessed: 2023-06-16.

[91] Qt Remote Objects. https://doc.qt.io/qt-6/qtremoteobjects-index.html. Accessed: 2023-05-31.

[92] QStandardItemModel Class. https://doc.qt.io/qt-6/qstandarditemmodel.html. Accessed: 2023-06-01.

[93] Property Binding. https://doc.qt.io/qt-6/qtqml-syntax-propertybinding.html. Accessed: 2023-08-15.

[94] Fleiss J.L., Levin B. & Paik M.C. (2013) Statistical methods for rates and proportions. John Wiley Sons, Hoboken, New Jersey, US.

[95] AUTOSAR C++ Rules and Coding Standards Compliance. https://www.parasoft.com/solutions/autosar/. Accessed: 2023-10-27.

[96] Axivion - Detecting unreachable and dead code. https://www.axivion.com/en/products/code-smells-detector/detecting-unreachable-code/. Accessed: 2023-10-24.

[97] Rule Of Three in C++. https://www.geeksforgeeks.org/rule-of-three-in-cpp/. Accessed: 2023-10-26.

[98] C++ Rule Of Three. https://www.tutorialspoint.com/cplusplus-rule-of-three. Accessed: 2023-10-26.

[99] Cohen's Kappa Calculator. https://www.graphpad.com/quickcalcs/kappa1/. Accessed: 2023-10-23.

[100] Landis J.R. & Koch G.G. (1977) The measurement of observer agreement for categorical data. Biometrics 33, pp. 159–174. URL: http://www.jstor.org/stable/2529310.

[101] Siedler C., Glatt M., Weber P., Ebert A. & Aurich J.C. (2021) Engineering changes in manufacturing systems supported by ar/vr collaboration. 8th CIRP Global Web Conference – Flexible Mass Customisation (CIRPe 2020) 96, p. 307–312.

[102] Valamede L.S. & Akkari A.C.S. (2020) Lean 4.0: A new holistic approach for the integration of lean manufacturing tools and digital technologies. International Journal of Mathematical, Engineering and Management Sciences 5.

[103] Hernandez-de Menendez M., Díaz C.A.E. & Morales-Menendez R. (2020) Engineering education for smart 4.0 technology: a review. International Journal on Interactive Design and Manufacturing (IJIDeM) 14, pp. 789–803.

[104] SonarQube. https://www.sonarsource.com/products/sonarqube/. Accessed: 2023-10-24.

[105] CodeScene. https://codescene.com. Accessed: 2023-10-24.

[106] Coverity. https://www.synopsys.com/software-integrity/security-testing/static-analysis-sast.html. Accessed: 2023-10-24.

[107] Message Queuing Telemetry Transport (MQTT). https://mqtt.org/. Accessed: 2023-09-4.

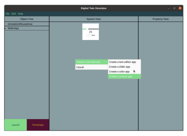
[108] The WebSocket API (WebSockets). https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API. Accessed: 2023-09-6.

[109] C++ Core Guidelines. https://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines. Accessed: 2023-10-27.

[110] MISRA C++. https://misra.org.uk/misra-c-plus-plus/. Accessed: 2023-10-27.

9. APPENDICES

Note 1: The Appendix 1 is an imported pdf file so the format looks different to the rest of the thesis.
Note 2: The Appendix 3 consists of images imported from the GraphPad’s Cohen’s Kappa calculator [99].



Digital Twin orchestrator UI study

This questionnaire is part of my master's thesis study. Please answer with as much detail and information to every question as you can. Your information is essentially valuable to my thesis work. Thank you.

* Required

This section is intended to inform you about how your results and information will be used or disclosed in the study. Your information will be used in accordance with this authorization form and the informed consent form as required or allowed by Finnish and EU laws. Please read the following carefully before accepting this form.

1. You do not have to accept this form
2. If you decide to participate, you are free to withdraw your authorization regarding the use and disclosure of the results information (and to discontinue any other participation in the study), except to the extent that the law allows us to continue using your information (e.g., necessary to maintain integrity of research).
3. Your observations, comments and information are pseudonymized, so that the data cannot be linked to your person.
4. The following students from the University of Oulu are authorized to use your results information in connection with this research study as described above: Juho Annunen - jannunen18@student.oulu.fi

1

Do you accept these terms? *

◯ I accept

◯ I do not accept

Your information

2

Sex *

○ Male

○ Female

○ Other

3

Your education *

4

What kind of professional background do you have? What kind of work do you do currently? Try to answer in a general level. Please don't provide any classified information. *

5

How many years have you been in your current profession? *

Digital Twin questions

6

Describe a Digital Twin using your own words. What are they used for? You can give some example(s) of a Digital Twin. *

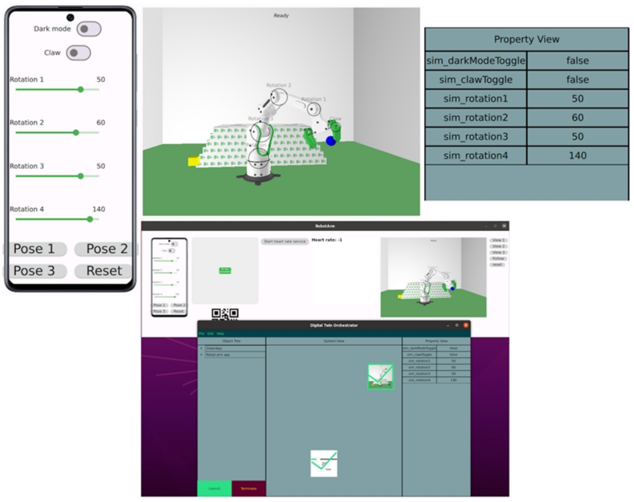
7

What current technologies would you integrate into this solution or for digital twin interfaces in general, please elaborate. (Examples: generative AI, machine learning, augmented reality, virtual reality, blockchain, 5G/6G, cloud/edge computing etc.) *

8

Do you think we need more solutions for interfacing with digital twins? Why? *

The demo



Property View	
sim_darkModeToggle	false
sim_clawToggle	false
sim_rotation1	50
sim_rotation2	60
sim_rotation3	50
sim_rotation4	140

9

What were your impressions about the demo?

10

Do you think that the Digital Twin orchestrator has a potential to be futher developed?

11

What additional functionalities would be needed for this type of digital twin interface? *

12

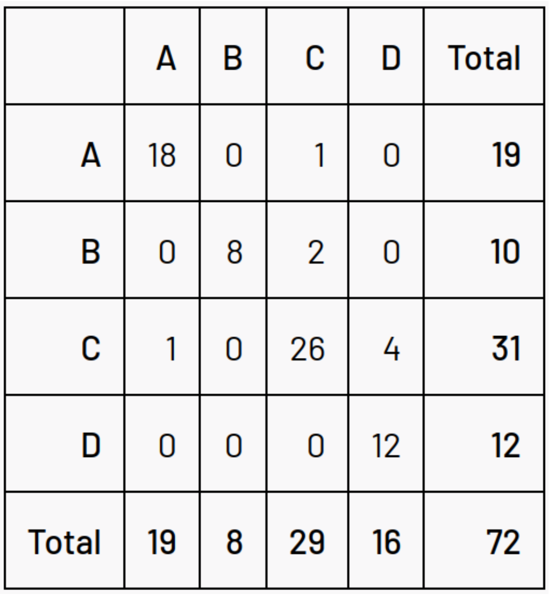
Can you give some other future development ideas for the Digital Twin orchestrator?

13

In what kind of situation and where in you current work would you use Digital Twin orchestrator or a similiar tool if you had it available? *

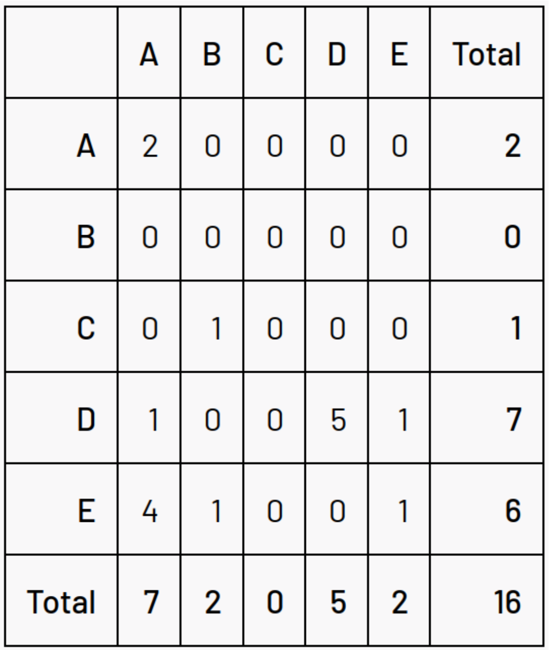
Table 3. Coding scheme

Theme	Codes
General feedback (16)	• Requires further development (7) • Easy to use (2) • Configurable (0) • Good technical demo (5) • Other (2)
Variability and abundance of solutions (8)	• Interoperability (1) • Multiplatform (5) • Integrated UI (1) • Standards and directives (1) • Other (2)
Potential and development targets (29)	• Automation (3) • AR (4) • Edge computing (2) • Analyzing capabilities (1) • clear visualization of connectivity (2) • launch Qt applications in an emulated Android device (1) • integration to Qt APIs (2) • Connect and control multiple devices (virtual and physical) (2) • UI features (3) • Other (14)
Use cases (19)	• building demos (1) • connecting multiple devices (2) • industrial automation applications (4) • Customer demos demonstrating Qt technology (5) • prototyping (4) • Other (4)



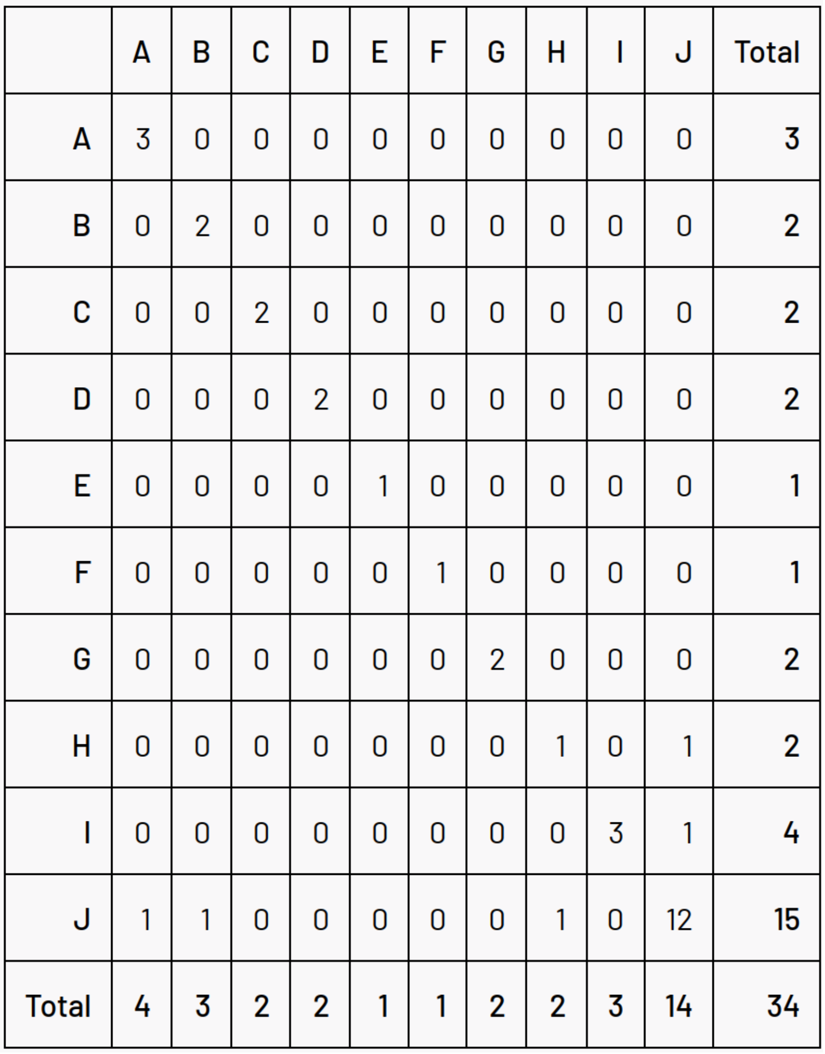
	A	B	C	D	Total
A	18	0	1	0	**19**
B	0	8	2	0	**10**
C	1	0	26	4	**31**
D	0	0	0	12	**12**
Total	**19**	**8**	**29**	**16**	**72**

Figure 16. Cohen's Kappa agreement table on themes (CC BY 4.0 Author)



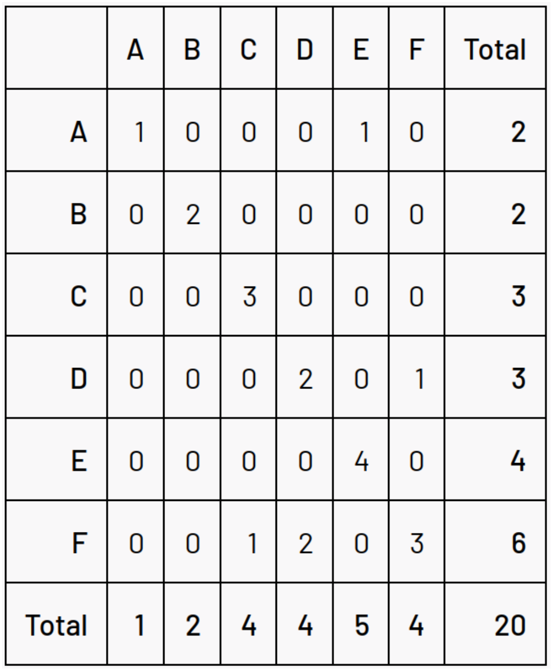
	A	B	C	D	E	Total
A	2	0	0	0	0	**2**
B	0	0	0	0	0	**0**
C	0	1	0	0	0	**1**
D	1	0	0	5	1	**7**
E	4	1	0	0	1	**6**
Total	**7**	**2**	**0**	**5**	**2**	**16**

Figure 17. Cohen's Kappa agreement table on general feedback (CC BY 4.0 Author)



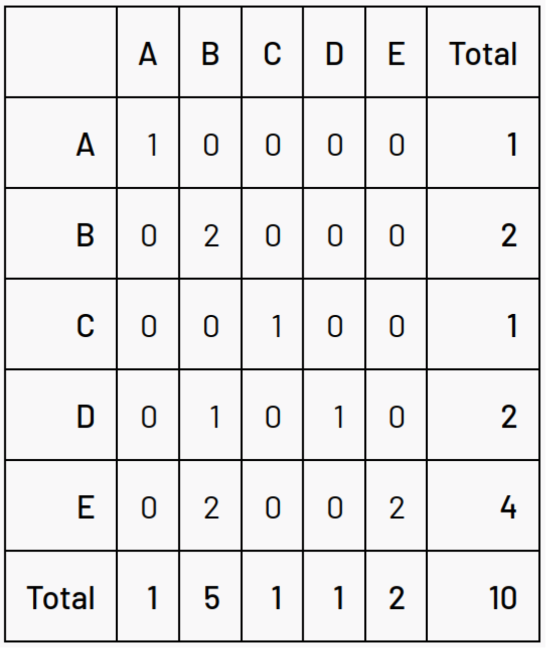
	A	B	C	D	E	F	G	H	I	J	Total
A	3	0	0	0	0	0	0	0	0	0	**3**
B	0	2	0	0	0	0	0	0	0	0	**2**
C	0	0	2	0	0	0	0	0	0	0	**2**
D	0	0	0	2	0	0	0	0	0	0	**2**
E	0	0	0	0	1	0	0	0	0	0	**1**
F	0	0	0	0	0	1	0	0	0	0	**1**
G	0	0	0	0	0	0	2	0	0	0	**2**
H	0	0	0	0	0	0	0	1	0	1	**2**
I	0	0	0	0	0	0	0	0	3	1	**4**
J	1	1	0	0	0	0	0	1	0	12	**15**
Total	**4**	**3**	**2**	**2**	**1**	**1**	**2**	**2**	**3**	**14**	**34**

Figure 18. Cohen’s Kappa agreement table on potential and development targets (CC BY 4.0 Author)



	A	B	C	D	E	F	Total
A	1	0	0	0	1	0	2
B	0	2	0	0	0	0	2
C	0	0	3	0	0	0	3
D	0	0	0	2	0	1	3
E	0	0	0	0	4	0	4
F	0	0	1	2	0	3	6
Total	1	2	4	4	5	4	20

Figure 19. Cohen’s Kappa agreement table on use cases (CC BY 4.0 Author)



	A	B	C	D	E	Total
A	1	0	0	0	0	1
B	0	2	0	0	0	2
C	0	0	1	0	0	1
D	0	1	0	1	0	2
E	0	2	0	0	2	4
Total	1	5	1	1	2	10

Figure 20. Cohen’s Kappa agreement table on Variability and abundance of solutions (CC BY 4.0 Author)