DigitalEnergyTwin

Mock-Up and specifications AR

D7.2

DIGITAL ENERGY TWIN – OPTIMISED OPERATION AND DESIGN OF INDUSTRIAL ENERGY SYSTEMS

Project Number: 873599

eCall Number: 21527440

Date: 30.7.2022

ENERGIEFORSCHUNG (E!MISSION)

5. AUSSCHREIBUNG ENERGIEFORSCHUNG 2018

Report Prepared by:

Name: Nino Brugger (a), Andreas Künz (a), Sabrina Rosmann (a)
Company: (a) FH Vorarlberg

This project has received funding from the *Klima und Energiefonds* under grant agreement No 873599.

Content

1 Executive summary

The intention of this deliverable is to plan the specifications of an AR app and to create a mock-up for it. The application is supposed to provide additional data for manufacturing units. These data could be information about the unit itself, recorded or predicted energy data or simply a real-time visualisation of the robot with the data from the twin software by digifai in order to check whether the virtual unit state of the software matches the real state of the physical unit.

To achieve this, an Augmented Reality headset is used; in our case the Microsoft HoloLens 2 as mentioned in the deliverable D7.1. This device provides holograms or digital / virtual elements to be placed in the real world and hand tracking. The former will be used to display 3D models of the units and additional information, the latter will be used to interact with the application in an easy way. The correct positioning of the virtual objects in the real world is important, especially when trying to position a virtual representation of an object on top of its real, physical origin – like the manufacturing units. This will be done by using the Vuforia Engine and Image Targets.

The two main states of the application will be real-time data and historic data. Each of them will have its own specifications and requirements. The real-time data comes from the twin software while historical data (recorded or predicted data) will be stored in CSV files. When being in the twin / real-time state, the app needs a connection to the twin servers, so the computer running the twin software and the HoloLens need to be in the same wi-fi network.

2 Mock-Ups of AR Visualizations

The AR application needs to provide the user with information about the energy state of the manufacturing units. In addition to this, useful information that is not related to energy data can be displayed (e.g., the angle of robot-arm units etc.).

To place such information in a useful manner in the environment, the AR application needs to know where the corresponding manufacturing unit is in the real world. Via markers the exact position of the environment can be tracked and therefore the exact location of the manufacturing unit can be calculated. In general, diagrams and flowcharts or similar information do not need to be placed anywhere on an existing object in the real world but can float around in front of the user. However, if there is data specifically for one manufacturing unit or a part of a manufacturing unit (such as angles of robot-arms etc.), it should be located where it applies.

The user should have the possibility to look at current and historical energy data. There are different options for how the historical data could be presented. It could be split up by hours, days, weeks or any other time frame or it could be split up by different production jobs. For the visualization of the real-time energy data, it could be useful to show the current state of the manufacturing unit as well.

3 Specifications of AR environment and functionalities

There are a number of tracking options, including plane, space, object and image tracking. For the purposes of this application, image tracking seems to be the best solution as it provides very precise and stable tracking without the need for complex learning procedures for room or object recognition.

To be able to interact with the application the HoloLens hand tracking and a simple menu should be used. Interactions with gestures that are too complicated (for example requiring precise movements with both hands at the same time) should be avoided as it needs to be usable by people that have never put on a HoloLens 2 as well. The HoloLens 2 also provides eye tracking, but this form of interaction is rather limited and seems to be not very intuitive or practical at the moment.

4 Specifications of AR interface, software, and hardware

4.1 Different App States

The two main types of information the user has to be able to see are real-time data and historic data. The historic data is divided into predicted data and recorded data. The app states are built to support exactly this structure.

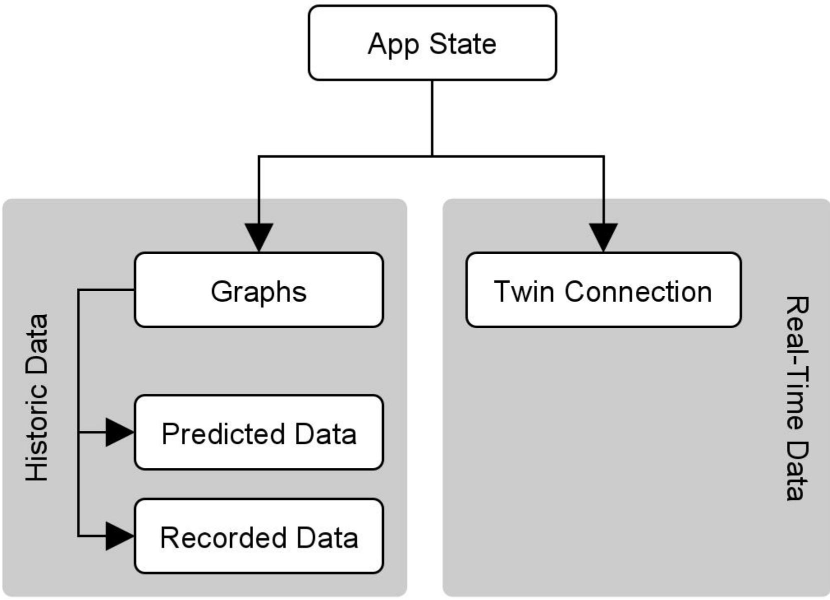

Figure 1: App States.

4.1.1 Real-Time Data

In the real-time state, the application will handle 3D model data from the Twin software and represent / build them in HoloLens application, respectively on the HoloLens. Therefore, a constant connection to the Twin server is required! The HoloLens 2 and the computer that the twin software is running on need to be in the same wi-fi.

The Start method of the ApplicationManager is the entry point into the application. The Update method is running each frame which is supposed to be at 60 times per second but can vary depending on the performance of the application.

The VisualisingController is responsible for visualising both the twin's 3D model data and machine data (additional information about parameters or states regarding the twin). It uses DataProviders for fetching the data and a PositionProvider for determining the position, the scale, and the rotation for placing these data correctly in physical space. ModelData objects contain the information that is needed for displaying 3D models. They hold MeshData objects with Triangles, Vertices, and Normals and a TransformationMatrix with the position, rotation, and scale of the model. MachineData objects contain the information associated with one related machine data set, including its position within the 3D model.

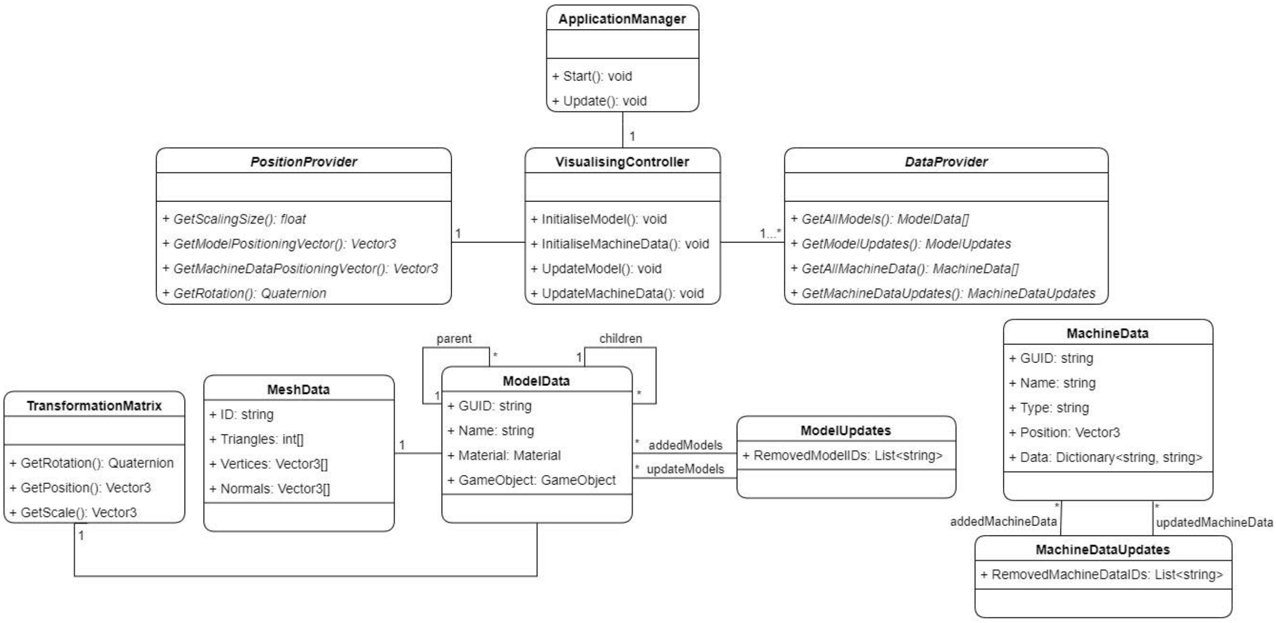

Figure 2: Architecture for the Twin connection.

4.1.2 Historic Data

The historic data can be devided into two types, which are the recorded data and the predicted data. Independently of the type of data, there needs to be a way to show and visualise the data. Because of this, a Model-View-Controller (MVC) architecture will be used to represent the data.

As shown in Figure 3, there are a few groups in this architecture. There is one group that takes care of the main graph systems. This group consists of the Graph2DController and the Graph2DLineView and -InformationView. The Controller handles the logic while the Views take care of representing and visualising everything. The Model of this MVC system is the CSV_GraphReader (could later be adapted to support other data formats than CSV). This class reads the data and arranges them in a useful way for the Controllers. Another group is the Range group which consists of an IntRangeVariable and a RangeSlider. This class is responsible for letting the viewer select a range of the graph so that he may “zoom in” or “zoom out” of the graph. The last group is responsible for animating the production unit according to the recorded data. In the beginning this will be a Kuka robot that’s why the elements are named accordingly – however, in the end the type of unit doesn’t matter and a robot or machine at AT&S can also be integrated without any additional effort.

The arrows show the dependencies, meaning that for example the KukaAnimationView is dependent on the KukaAnimationController. It is important that the dependencies flow into the right direction. For instance, nothing is dependent on the Kuka Animation group, which allows graphs to be made without any Kuka animations. There are also nowhere any circular dependencies.

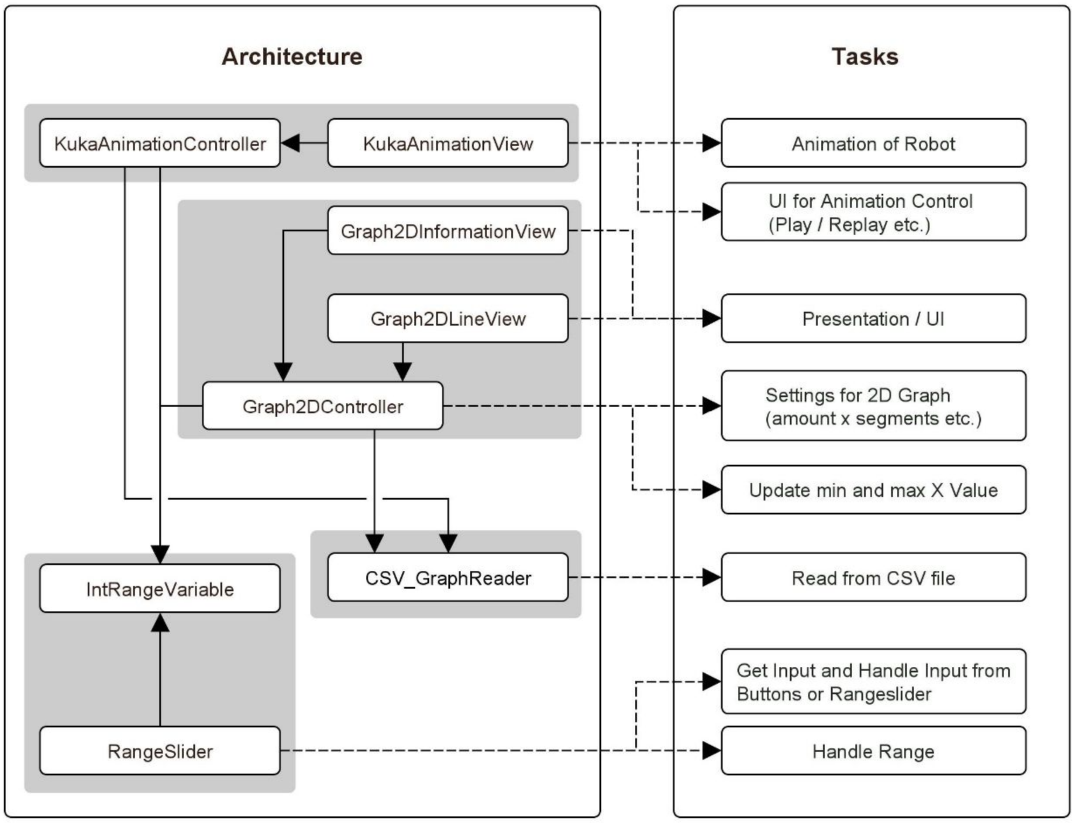

Figure 3: Graph Architecture.

4.2 Development

For the development of the application Unity and the Mixed Reality Toolkit will be used. Unity is a game engine that offers game/application production for a lot of different platforms, one of which is the Universal Windows Platform. This is what is needed for the Microsoft HoloLens 2. The Mixed Reality Toolkit is a plugin or package for the Unity engine, which allows us to use a lot of the feature the HoloLens offers, e.g. the hand tracking. The software twin by digifai is used as the data source for the Digital Twin data. Within the twin software, a Digital Twin of the industrial learning factory Digital Factory at the University of Applied Sciences Vorarlberg is running. Next to the 3d object information of the production line, real-time energy data will be provided by the twin software, too. However, for historical data importing database tables via csv is planned. For positioning the visualisations in physical space the Vuforia plugin is used, which is an Augmented Reality software development kit for marker-based tracking in real-time with images, QR codes or simple 3D objects as markers. The Digital Twin visualisations (real-time data) are anchored in physical space with three Vuforia image targets, which are distributed all around the Digital Factory pipeline. The graphs for the historical data will be anchored using only two markers directly next to one of the manufacturing units.

5 Summary and conclusion

The AR-application will be built with the Unity Engine for the HoloLens 2 from Microsoft and will be used to view information about manufacturing units and their energy-related data. The two main states of this app are real-time data and historical data. The real-time data state uses a connection to the Twin software by digifai to represent the real-time state of Digital Factory in the University of Applied Sciences Vorarlberg. The placement of the virtual factory is achieved via image markers. This state is not very interactive as it is supposed to show real-time data. The historical data on the other hand is supposed to be rather interactive; different types of graphs are shown and can be adjusted, zooming in or changing the region that it shows.