A logo with blue circles and black text

Description automatically generated

|  |
| --- |
| Technische Hochschule Ulm |
| Digital Twin of Kuka KR3 |
| System Requirements |

|  |
| --- |
| Ibrahim Almohamed, Ahmed  21.08.2024 |

Contents

[1. Version and Control 2](#_Toc176737843)

[2. Glossary 3](#_Toc176737844)

[3. System Overview 5](#_Toc176737845)

[4. System Components 6](#_Toc176737846)

[5. App Sketching 7](#_Toc176737847)

[a. LoginScreen 7](#_Toc176737848)

[b. HomeScreen 8](#_Toc176737849)

[c. DevicesScreen 9](#_Toc176737850)

[d. ConnectionScreen 11](#_Toc176737851)

[e. SimulationScreen 12](#_Toc176737852)

[6. System Use cases 14](#_Toc176737853)

[7. System Requirements 25](#_Toc176737854)

[a. Functional Requirements 25](#_Toc176737855)

[i. ROS2-KR3 Core 25](#_Toc176737856)

[ii. OPCUA-Server 25](#_Toc176737857)

[iii. Gazebo-Simulation 25](#_Toc176737858)

[iv. Database 25](#_Toc176737859)

[v. Dashboard 25](#_Toc176737860)

[vi. KR3-R540 25](#_Toc176737861)

[b. Nonfunctional requirements 25](#_Toc176737862)

[8. Templates 26](#_Toc176737863)

# Version and Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Name of Editor | Notes | Date |
| 1.0.0 | Ahmed Ibrahim Almohamed | Related to CustomerRequirementV1\_0\_0.docx | 21.08.2024 |
| 1.0.1 | Ahmed Ibrahim Almohamed | Update : System concept and overview and added initial system requirements | 09.09.2024 |
|  |  |  |  |

# Glossary

|  |  |
| --- | --- |
| Term | Description |
| DT | Digital Twin |
| KukaDigitalTwin | A Digital twin system of the Kuka KR3 using ROS and Gazebo (simulation tool) . |
| KukaVerse | A Dashboard subsystem that creates a OPCUA server and control the DT . |
| ROS2KR3Core | A software based on ROS2 that runs and manages the simulation of the KR3 Digital twin. |
| ROS2KR3Connection | A connection method to connect the ROS2 (or the host PC) with the KUKA KR3 robot using KUKAVARPROXY . |
| ROS2OPCUABridge | a Software bridge to map a ROS2 node to a OPCUA node. |
| AKL | “Automatisches Kleinteilelager” (DE) or “Automated small parts warehouse” (EN) |
| ROS | Robot Operating System |
| Kuka KR3 | A robotic arm with a microscope  Description automatically generated |
| KVP | KUKAVARPROXY |
| OPC-UA |  |
| SoftRealTime | system where deadlines are important but missing them occasionally does not result in system failure.(average delay of 5ms-30ms) |
| BiDirectionConnection | A connection between the physical and digital robots where commands can be sent from either robot to control the other, and the state information (such as position, velocity, sensor data, etc.) is continuously exchanged. |
| MoveIt2 | A robotic manipulation platform for ROS 2 and incorporates the latest advances in motion planning, manipulation, 3D perception, kinematics, control, and navigation |
| RosInterface | A software interface for the Ros2 to connect the Controllers and the simulation of Gazebo with the KVP protocol. |
| GUI | Graphical User Interface |
| RoboticsLab | A Laboratory at the THU that is used for running experiments of robotics. |
| KukaDigitalTwinDashboard | A Dashboard which is a part of the KukadigitalTwin GUI , used for control and monitor the digital twin and the real twin. |
| RosTasks | A RosTask is a software that aims to create a simple or complicated task for the KukaDigitalTwin , where the user writes a RosNode ,that is runnable on both the physical and digital twins. |
| RQT | RQT is a graphical user interface (GUI) tool for ROS 2. Everything done in RQT can be done on the command line, but RQT provides a more user-friendly way to manipulate ROS 2 elements. |
| RosNode | A node is a participant in the ROS 2 graph, which uses a client library to communicate with other nodes. Nodes can communicate with other nodes within the same process, in a different process, or on a different machine. Nodes are typically the unit of computation in a ROS graph; each node should do one logical thing. |

# System Overview

A diagram of a computer system

Description automatically generated with medium confidence

Figure 1: System Overview

This system overview of the KukaDigitalTwin illustrates the integration of a **KR3-R540 robot** using **ROS2**, **MoveIt 2**, and **Gazebo Simulation** with an **OPC UA** interface for remote control and monitoring. At the core of the system is the **ROS2-KR3 Core**, which manages the real-time control of the robot. The core includes ROS2 components such as **MoveIt 2** for motion planning, **ROS2 controllers** for managing the robot's movements, and interfaces for communicating with both the real robot and the **Gazebo simulation** (acting as a digital twin).

High-level commands are issued from a **dashboard** interface, where users can interact with the system remotely. These commands are communicated via an **OPC UA server**, which acts as an intermediary between the dashboard and the ROS2 core. The **OPC UA client** within the ROS2 core receives these commands, which are then translated into real-time instructions for the robot. The **KR3-KUKAVARPROXY** handles direct communication with the physical **KR3-R540 robot** and its **KRC4 controller** via a TCP connection.

The **database** component stores user and system data, ensuring that the system state is maintained and accessible for monitoring and further control tasks. Overall, this architecture ensures that high-level task execution and real-time robot control are seamlessly integrated, allowing for efficient remote operation and simulation-based testing.

In this Document it will defined the System use cases and the system requirements ,due to the System-Engineering approach of developing this system .

# System Components

At the core is the **ROS2-KR3 Core**, responsible for managing real-time control. This core contains the **KR3-Description**, which defines the robot’s model and configuration, as well as **KR3-MoveIt2** for motion planning and path execution. Additionally, the core houses **ROS2 controllers** that handle specific control tasks like joint movements and trajectory following. Interfaces such as the **KR3 Ros2-GazeboInterface** link the core to the Gazebo Simulation, enabling real-time testing in a digital twin environment, while the **KR3 Ros-OpenShowVarInterface** facilitates communication with the physical robot via the KR3-KUKAVARPROXY, which bridges to the KR3-R540 robot and its KRC4 Controller over TCP.

The **OPC UA Server** plays a crucial role in enabling remote interaction. It facilitates the exchange of high-level commands and system state data between the **Dashboard** (used by local and remote operators) and the **ROS2-KR3 core**. The **OPC UA Client embedded in the ROS2** core communicates directly with the OPC UA server, processing incoming commands and forwarding them to the appropriate ROS2 components for real-time execution. Meanwhile, the **Ros2-Controlles** ensures that both the physical robot and the digital twin receive the same commands, allowing for synchronized testing and real-world execution. **The SQL Database** stores system and user data, providing persistence and access to critical information for monitoring and control.

Altogether, this setup supports a complete workflow from remote command issuance via OPC UA to real-time robot control, enhanced by simulation capabilities using Gazebo.

These components will be discussed in more detail when the system reaches the fine design phase of the project .

# App Sketching

## LoginScreen

A screenshot of a login form

Description automatically generated

Figure 2: KukaVerse:Login

This is the start point of the KukaVerse , where the use can login or reset the password.

## HomeScreen

A screenshot of a computer

Description automatically generated

Figure 3: KukaVerse:Home

This is the HomeScreen of the KukaVerse , where the user can see some information about the system.

On the side there is also the is the sections of the app , the state of app connection and the username of the current user.

The Simulate button is disabled here until a connection to the robot is established.

## DevicesScreen

A screenshot of a computer

Description automatically generated

Figure 4: KukaVerse: Devices

The DevicesScreen is the main screen where all the current users robots and their configurations are displayed .

The user can also make new entries here by clicking on the “+” icon in the bottom right corner

.

Which leads to the AddDeviceScreen

A screenshot of a computer

Description automatically generated

Figure 5: KukaVerse : Add Device

On this screen the user can create a new robot with a description , type and name (and more configurations are TBD).

The user saves the new robot so it will be displayed as a card on the DevicesScreen.

## ConnectionScreen

A screenshot of a computer

Description automatically generated

Figure 6: KukaVerse : Connection

On the ConnectionScreen the user can configure the connection to the OPCUA server ,that starts when the connect button is pressed.

The user can set a Device( from stored devices) ,an Endpoint URL and a logging level.

After a connection is established the State “Not connected” changes to “Connected” and the simulate button is unlocked.

## SimulationScreen

A screenshot of a computer

Description automatically generated

Figure 7: KukaVerse : Simulate

On the SimulationScreen , The user views the device and connection configurations.

After that the user Starts the Simulation by pressing the start simulation button that starts the Gazebo on another window and starts the KukaVerse control panel in single mode.

A screenshot of a computer

Description automatically generated

Figure 8: KukaVerse : Control panel :Single

The User here can set a point (x,y,z) and a speed and click on the Go to Point button (the robot here shall start moving to that point and also the simulation).

The user can also check the Set Multi-Points option to save a set of point the robot reach (point1 -> point2 -> point3 ….) after setting each point by clicking the set point Button , and starting a robot job with go to point.

A screenshot of a computer

Description automatically generated

Figure 9: KukaVerse : Control panel Multi

# System Use cases

A diagram of a computer

Description automatically generated

Figure 10: System Use cases

|  |  |
| --- | --- |
| **Identifier** | **UC1** |
| **Name** | LoginMode |
| **Description** | The KukaVerse logs the user in |
| **Preconditions** | The KukaVerse is started |
| **Postconditions** | The KukaVerse is starts on UC2:HomeMode  OR  The KukaVerse is on UC1:LoginMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User starts the KukaVerse app |
| **Standard Procedure** | * 1. The KukaVerse starts the LoginScreen   2. The User inputs the UserEmail and UserPassword   3. The User presses the LoginButton   4. The KukaVerse displays a “LoginSuccessfulMessage” for 3 seconds and continues with UC2: HomeMode |
| **Alternative Procedure** | 4a.1 The KukaVerse displays “LoginFaildMessage” for 3 seconds    4a.2 The KukaVerse continues with UC1:LoginMode |

|  |  |
| --- | --- |
| **Identifier** | **UC2** |
| **Name** | HomeMode |
| **Description** | The KukaVerse displays the HomeScreen and the MainDashboard |
| **Preconditions** | The User logs in successfully |
| **Postconditions** | The KukaVerse is in UC3: DevicesMode  OR  The KukaVerse is in UC5:ConnectionMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User logs in successfully |
| **Standard Procedure** | 1. The KukaVerse displays KukaVerseInfo 2. The KukaVerse displays the list of KukaVerseFunctionalityList and “SystemNotConnectedMessage” 3. The User selects “Devices” form the KukaVerseFunctionalityList then the KukaVerse continues with UC3:DevicesMode |
| **Alternative Procedure** | 3a The User selects “Connect” from the KukaVerseFunctionalityList then the KukaVerse continues with UC4:ConnectionMode |

|  |  |
| --- | --- |
| **Identifier** | **UC3** |
| **Name** | DevicesMode |
| **Description** | The KukaVerse displays the DevicesList |
| **Preconditions** | The KukaVerse is in UC2:HomeMode  OR  The KukaVerse is in UC5:ConnectionMode |
| **Postconditions** | The KukaVerse is in UC2: HomeMode  OR  The KukaVerse is in UC4:AddDeviceMode  OR  The KukaVerse is in UC5:ConnectionMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User selects “Devices” from the KukaVerseFunctionalityList |
| **Standard Procedure** | 1. The KukaVerse displays all stored DeviceCard on DevicesScreen 2. The KukaVerse displays the list of KukaVerseFunctionalityList and “SystemNotConnectedMessage” 3. The KukaVerse displays AddDeviceButton 4. If the User pushes the AddDeviceButton then the KukaVerse continues with UC4:AddDeviceMode |
| **Alternative Procedure** | 1a. If there is no DeviceCard stored then the KukaVerse displays DefaultDeviceCard  4a. If the User selects “Connect” from the KukaVerseFunctionalityList then the KukaVerse continues with UC4:ConnectionMode  4a. If the User “Home” from the KukaVerseFunctionalityList then the KukaVerse continues with UC2:HomeMode |

|  |  |
| --- | --- |
| **Identifier** | **UC4** |
| **Name** | AddDeviceMode |
| **Description** | The User adds and stores a new DeviceCard . |
| **Preconditions** | The KukaVerse is in UC3:DevicesMode |
| **Postconditions** | The KukaVerse is in UC3:DevicesMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User pushes The User pushes the AddDeviceButton on the DevicesScreen |
| **Standard Procedure** | 1. The KukaVerse displays the NewDevicePannel on AddDeviceScreen 2. The User inputs a NewDeviceName 3. The User selects a NewDeviceType 4. The User inputs a NewDeviceDescription 5. If the User selects SaveDeviceButton then the KukaVerse stores the NewDevice and continues with UC3: DevicesMode with a new DeviceCard on the DevicesScreen. |
| **Alternative Procedure** | 5a. If the User selects CancelButton then the KukaVerse continues with UC3:DevicesMode with no NewDevice and no new DeviceCard on the DevicesScreem |

|  |  |
| --- | --- |
| **Identifier** | **UC5** |
| **Name** | ConnectionMode |
| **Description** | The KukaVerse displays the ConnectionPannl on the ConnectionScreen |
| **Preconditions** | The KukaVerse is in UC2:HomeMode  OR  The KukaVerse is in UC3:DevicesMode |
| **Postconditions** | The KukaVerse is in UC6:SimluationMode  OR  The KukaVerse is in UC3:DevicesMode  OR  The KukaVerse is in UC2:HomeMode |
| **Failure Scenarios** | KukaVerse fails to connect the KukaVerseOpcuaClient to the KukaOpcuaServer. |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User selects “Devices” from the KukaVerseFunctionalityList |
| **Standard Procedure** | 1. The KukaVerse displays the ConnectionScreen with ConnectionPannel 2. The User selects the Device to be connected 3. The User inputs the EndpointURL 4. The User selects the LoggingLevel of the connection 5. If The User pushes the ConnectButton then the KukaVerse start the following:    1. Connects the KukaVerseOpcuaClient to the KukaOpcuaServer    2. Displays a “SystemConnectedMessage”    3. Enables the “Simulate” option on the KukaVerseFunctionalityList 6. If the User selects “Simulate” from the KukaVerseFunctionalityList then the KukaVerse continues with UC6:SimulationMode |
| **Alternative Procedure** | 5a if KukaVerse fails to connect the KukaVerseOpcuaClient to the KukaOpcuaServer then the KukaVerse starts the following :   * 1. displays “OpcuaConnectionErrorMessage”   2. displays “SystemNotConnectedMessage”   3. disables the “Simulate” option on the KukaVerseFunctionalityList if it was enabled   6a If the User “Home” from the KukaVerseFunctionalityList then the KukaVerse continues with UC2:HomeMode  6a if the User selects “Devices” form the KukaVerseFunctionalityList then the KukaVerse continues with UC3:DevicesMode |

|  |  |
| --- | --- |
| **Identifier** | **UC6** |
| **Name** | SimulationMode |
| **Description** | The KukaVerse displays the selected DeviceCard and the ConnecctionCard |
| **Preconditions** | The KukaVerse displays SystemConnectedMessage |
| **Postconditions** | The KukaVerse is in UC7:ControlPanelMode  OR  The KukaVerse is in UC5:ConnectionMode  OR  The KukaVerse is in UC:3DevicesMode  OR  The KukaVerse is in UC2:HomeMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User selects “Simulate” from the KukaVerseFunctionalityList |
| **Standard Procedure** | 1. The KukaVerse displays the following :    1. DeviceCard about the selected Device    2. ConnectionCard about the configured connection. 2. If the User pushes the StartSimulationButton then the KukaVerse continues with UC7:ControlPanelMode |
| **Alternative Procedure** | 2a if the User selects “Devices” form the KukaVerseFunctionalityList then the KukaVerse continues with UC3:DevicesMode  2a if the User selects “Connect” form the KukaVerseFunctionalityList then the KukaVerse continues with UC5: ConnectionMode  2a if the User selects “Home” form the KukaVerseFunctionalityList then the KukaVerse continues with UC2:HomeMode |

|  |  |
| --- | --- |
| **Identifier** | **UC7** |
| **Name** | ControlPanelMode |
| **Description** | The KukaVerse displays ControlPannel on the ControlPannelScreen |
| **Preconditions** | The user pushes the “StartSimualtionButton” in UC6:SimulationMode |
| **Postconditions** | The KukaVerse is in UC7: ControlPanelMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User pushes the StartSimulationButton on SimulationScreen in UC6:SimulationMode |
| **Standard Procedure** | 1. The KukaVerse displays the ControlPannel 2. The User sets the following of the PointPannel:    1. The RobotSpeed    2. The GoToPointXCoordinate    3. The GoToPointYCoordinate    4. The GoToPointZCoordinate 3. The User pushes the GoToPointButton 4. The KukaVerse sends GoToPointCommand to the KukaOpcuaServer 5. The KukaVerse displays the following :    1. The RobotStateCard    2. The SimulationStateCard    3. A DefaultPointsCard 6. The KukaVerse continues with UC7: ControlPanelMode |
| **Alternative Procedure** | 1a. The User checks the SetMultiPointsOption  3a. The User pushes the SetPointButton and continues on step 2 of the Standard Prorcedure (steps 1a and 3a repeated for maximum of 4 points)  5a. The KukaVerse displays the following :   * 1. The RobotStateCard   2. The SimulationStateCard   3. The PointToReachCard |

|  |  |
| --- | --- |
| **Identifier** | **UC8** |
| **Name** | LogoutMode |
| **Description** | The KukaVerse logs the user out |
| **Preconditions** | The KukaVerse is in any state |
| **Postconditions** | The KukaVerse is in UC1:LoginMode |
| **Failure Scenarios** | n/a |
| **Related Usecases** | n/a |
| **Actors** | User |
| **Trigger** | The User selects the “Logout” option on the KukaVerseFunctionalityList |
| **Standard Procedure** | 1. The KukaVerse stops any running simulations 2. The KukaVerse disconnects the KukaVerseOpcuaClient from the KukaOpcuaServer 3. The KukaVerse dispalys “LogoutMessage” 4. The KukaVerse continues with UC2:LoginMode |
| **Alternative Procedure** | n/a |

# System Requirements

## Functional Requirements

### ROS2-KR3 Core

|  |  |
| --- | --- |
| Requirement ID | SR-1 ( this should be non functional! ) |
| digital model | The KukaDigitalTwin shall have a viewable and interactable 3d Model using URDF or XACRO. |

|  |  |
| --- | --- |
| Requirement ID | SR-1 |
| Plan  Motion | In the context of ROS2-KR3 Core the KukaDigitalTwin shall plan the KR3-R540 and the Gazebo-Simulation using Moveit2 Framework of the ROS2. |

|  |  |
| --- | --- |
| Requirement ID | SR-2 |
| Control Robot | In the context of ROS2-KR3 Core the KukaDigitalTwin shall control the real robot (e.g KR3-R540) using the Ros-OpenShowVarInterface . |

|  |  |
| --- | --- |
| Requirement ID | SR-3 |
| Control Simulation | In the context of ROS2-KR3 Core the KukaDigitalTwin shall control the DigitalTwin (in the Gazebo Simulation) using the Ros2-GazboInterface. |

|  |  |
| --- | --- |
| Requirement ID | SR-4 |
| OPCUA-Communication | In the context of ROS2-KR3 Core the KukaDigitalTwin shall connect the ROS2-KR3 Core with the OPCUA-Server by using a ROS-OPCUA Bridge. |

|  |  |
| --- | --- |
| Requirement ID | SR-5 |
| Start Task | In the context of ROS2-KR3 Core the KukaDigitalTwin shall start a task be receiving a StartTaskCommand from the ROS-OPCUA Bridge and executing the controllers (on simulation and robot). |

|  |  |
| --- | --- |
| Requirement ID | SR-6 |
| Run Task | In the context of ROS2-KR3 Core the KukaDigitalTwin shall run a task be receiving a RunTaskCommand from the ROS-OPCUA Bridge and executing the controllers (on simulation and robot). |

|  |  |
| --- | --- |
| Requirement ID | SR-7 |
| Stop Task  (This is Missing in the Usecases) | In the context of ROS2-KR3 Core the KukaDigitalTwin shall stop a task be receiving a StopTaskCommand from the ROS-OPCUA Bridge and executing the controllers (on simulation and robot). |

|  |  |
| --- | --- |
| Requirement ID | SR-8 |
| Pause Task  (This is Missing in the Usecases) | In the context of ROS2-KR3 Core the KukaDigitalTwin shall pause a task be receiving a StopTaskCommand from the ROS-OPCUA Bridge and executing the controllers (on simulation and robot). |

|  |  |
| --- | --- |
| Requirement ID | SR-9 |
| Resume Task  (This is Missing in the Usecases) | In the context of ROS2-KR3 Core the KukaDigitalTwin shall resumes a task be receiving a ResumeTaskCommand from the ROS-OPCUA Bridge and executing the controllers (on simulation and robot). |

|  |  |
| --- | --- |
| Requirement ID | SR-10 |
| sync Tasks | In the context of ROS2-KR3 Core the KukaDigitalTwin shall sync all tasks between the KR3-R540 and the Gazebo-Simulation. |

### OPCUA-Server

|  |  |
| --- | --- |
| Requirement ID | SR-11 |
| receive user commands | In the context of OPCUA-Server the KukaDigitalTwin shall receive user commands at the OPCUA-Server from the Dashboard. |

|  |  |
| --- | --- |
| Requirement ID | SR-12 |
| Receive  RosData | In the context of OPCUA-Server the KukaDigitalTwin shall receives RosData from the ROS2-KR3 Core. |

|  |  |
| --- | --- |
| Requirement ID | SR-13 |
| Send  HighLevelRosCommands | In the context of OPCUA-Server the KukaDigitalTwin shall send HighLevelRosCommands to ROS2-KR3 Core.. |

### Gazebo-Simulation

|  |  |
| --- | --- |
| Requirement ID | SR-14 |
| Start Simulation | In the context of Gazebo-Simulation the KukaDigitalTwin shall start the KR3-Simulation-world window when the ROS2-KR3 Core starts. |

|  |  |
| --- | --- |
| Requirement ID | SR-15 |
| Run Simulation | In the context of Gazebo-Simulation the KukaDigitalTwin shall run the KR3-Simulation when the ROS2-KR3 core runs a task. |

|  |  |
| --- | --- |
| Requirement ID | SR-16 |
| Stop Simulation  (This is Missing in the Usecases) | In the context of Gazebo-Simulation the KukaDigitalTwin shall stop the KR3-Simulation when the ROS2-KR3 core stops a task. |

|  |  |
| --- | --- |
| Requirement ID | SR-17 |
| pause Simulation  (This is Missing in the Usecases) | In the context of Gazebo-Simulation the KukaDigitalTwin shall pause the KR3-Simulation when the ROS2-KR3 core pasues a task. |

|  |  |
| --- | --- |
| Requirement ID | SR-18 |
| Resume Simulation  (This is Missing in the Usecases) | In the context of Gazebo-Simulation the KukaDigitalTwin shall resume the KR3-Simulation the ROS2-KR3 core resums a task. |

### Database

|  |  |
| --- | --- |
| Requirement ID | SR-19 |
|  | . |

### Dashboard

### KR3-R540

## Nonfunctional requirements

# Templates

|  |  |
| --- | --- |
| Requirement ID |  |
|  |  |

|  |  |
| --- | --- |
| **Identifier** |  |
| **Name** |  |
| **Description** |  |
| **Preconditions** |  |
| **Postconditions** |  |
| **Failure Scenarios** |  |
| **Related Usecases** |  |
| **Actors** |  |
| **Trigger** |  |
| **Standard Procedure** |  |
| **Alternative Procedure** |  |