Matas Führer, Roland Heinrich, Abdelwadoud Mabrouk, Tobias Christian Piller, Abdelmajid Khelil and Kubilay Yildiz

Online-MQTT: Online Lab for Basic and Advanced Features of the MQTT Protocol

Abstract

The Message Queuing Telemetry Transport (MQTT) application layer protocol is the de-facto first choice in building Internet of Things (IoT) applications. Accordingly, MQTT is being increasingly deployed and configured by experts as well as non-experienced application designers. Furthermore, MQTT is being increasingly used for Industrial IoT (IIoT) use cases, where even mission-critical measurements and commands are shared by machines through MQTT. Being a fundamental component in virtually all (I)IoT projects, a wrong MQTT configuration or use may lead to critical loss or damage. This remarkable popularity of MQTT has resulted in a tremendous need for problem- and task-driven MQTT training with hardware-in-loop, which well suits a wide range of qualification degrees. Precisely, application designers require a remote lab that is scalable, accessible anytime, and involves low-cost machine models. Consequently, we propose Online-MQTT, an online lab for MQTT with hardware-in-loop. Online-MQTT is designed to be used by a wide range of learners while pinpointing selected crucial research gaps and possible solution strategies for spontaneous and/or real-time MQTT communications for machines.

Keywords

Industrial Internet of Things (IIoT); Message Queuing Telemetry Transport (MQTT); Online-Lab-based IoT Education

1 Introduction

Message Queuing Telemetry Transport (MQTT) is a popular publish-subscribe messaging protocol in the Internet of Things (IoT). MQTT is frequently used by IoT system and application designers in virtually all application

domains, from smarthome to smart agriculture, smart city, Industrial IoT (IIoT), etc. As machines increasingly act as publishers and subscribers, spontaneous communication across machines, i.e., without an explicit pre-configuration at the commissioning phase, is required more and more in IIoT. In addition, the need for real-time communication via MQTT became apparent. Accordingly, relevant MQTT features need to be further developed and in particular made automated and machine-ready. MQTT is easy to use and is consequently being used and configured more and more, even by non-IT users. This makes MQTT a fundamental but critical component in IoT. A wrong configuration by a non-experienced user may lead to serious damage in mission-critical use cases. Therefore, it is essential to offer labs where participants from various education and technical levels can safely experiment with basic and advanced MQTT features. Obviously, such a lab needs to be low-cost, extensible, and trustworthy to scale globally.

Offering a physical MQTT-driven IoT lab requires remarkable personal and monetary efforts. Furthermore, these labs are per se not designed for remote access. Subsequently, they do not scale globally, though such a scale would make the efforts invested more justifiable. Accordingly, it is fundamental to digitalize these labs and offer them to remote users. Unfortunately, enabling meaningful remote access to a physical lab may also be time and cost-intensive.

Besides the publicly available specification of the MQTT standard, some online MQTT learning platforms exist. Some blogs such as (Eclipse Foundation, 2021a), (HiveMQ, 2021) and MQTT.org offer valuable tool recommendations and examples for self-learning. Furthermore, some online (mostly cloud-based) MQTT brokers and clients offer an infrastructure for quick trials with MQTT. In addition, some simulation environments exist, such as the MIMIC MQTT Simulator (Gambit Communications, 2021). Obviously, these resources are rather for experts and require a long self-learning phase for beginners. Additionally, they insufficiently pinpoint the limitations of the current MQTT standard and how to address them in research.

A few digital IoT labs already exist. The partners of the DigiLab4U Project (the DigiLab4U Project, 2021b) offer some of them. The FIT Lab (French Ministry of Higher Education, 2021) focuses on networking and wireless communication aspects. Unfortunately, these labs do not provide a digital lab on MQTT. Only a few online task-based MQTT courses exist, such as the hands-on introduction to MQTT (Manzoni, 2021). However, they do not provide access to hardware assets in a physical lab. To the best of our knowledge, there is no online lab that covers standardized as well as Work in Progress (WiP) features while being suited to all education

levels. We propose Online-MQTT, which represents our first steps towards a multi-purpose MQTT online lab.

Besides standardized features (client, broker, testbed), we decided to teach WiP features that enrich the semantic and real-time properties of MQTT so that designers become aware of the limitations which the current standard versions, MQTT 3.1.1 (Oasis, 2021a) and MQTT 5.0 (Oasis, 2021b), suffer from. These WiP features may inspire advanced learners for future business and research opportunities. DigiLab4U provides an appropriate infrastructure to enable educational remote access to physical IoT Labs. We are building on this opportunity to realize Online-MQTT lab.

The remainder of this paper is organized as follows. In Section 2, we detail the requirements of an Online-MQTT digital lab. The implementation of Online-MQTT and its integration into the open DigiLab4U infrastructure is discussed in Section 3. In Section 4, we revisit our requirements and qualitatively assess the quality of the proposed digital lab. Section 5 concludes the paper and gives a brief overview of our next steps.

2 Requirements

The key drivers to allow remote access to a given physical lab are (1) allowing current users to further use the lab anywhere and at any time, and (2) reaching as yet unreachable users worldwide.

As the number of unreachable users is unknown in advance and lab providers usually desire to maximize the worthiness of the digitalization of their labs through a larger number of users, scalability with regard to (simultaneous) number of users is a crucial requirement in the design of Online-MQTT.

In order to allow for scalability and simultaneity of use, it is essential to base the basic instance of Online-MQTT on low-cost, multi-purpose and Commercial-Off-The-Shelf (COTS) components. This would allow operating expenses to be reduced and new demands to be responded to very fast.

The hardware and software lab architecture should be legolized in such a way that some parts are easy to interchange, and some new requirements are easily implemented. The extensibility of the online lab is essential for its longevity and evolvability in the fast-growing (I)IoT world.

Finally, the ecosystem of the online lab should be trustworthy so that no significant harm is caused to any lab asset through remote access. In particular, the attack surface of the infrastructure should not be increased,

the privacy of lab users should be protected, and safety in the lab should be maintained.

3 Our Approach for an Online-MQTT Lab

Opening a physical lab to external users is usually challenging. Opening it for a remote user is even more challenging. This complexity depends directly on the experiment to be shared. In the introduction, we laid out several virtually simulated MQTT experiments a student could run on his/her own or on a provided machine. To further enrich the learning process, it was important to us that a student could experiment with physical instruments in our lab. A physical demo that can be accessed and experimented with in real-time, even remotely, should be a great didactic tool to help the student, through his/her own experimentation with the robots, observe and understand the limitations of MQTT and the need for its support of RT-protocols. With these considerations, the aim of this project is to emulate an IIoT scenario, where MQTT is used to transmit commands to machines. Our major steps toward an Online-MQTT lab that fulfills the requirements above consist in:

- Making use of the DigiLab4U infrastructure for COTS standard digitalization modules (Sec. 3.2).
- Utilizing the popular LEGO EV3 robots while enabling MQTT for them to realize a low-cost but powerful machine model (Sec. 3.3).
- Implementing a real-time loop for immersive user interaction, i.e., experiment control and result visualization (Sec. 3.4).
- Supporting simultaneous access to the lab through spatial and time redundancy considerations in the design (Sec. 3.5).

3.1 Technical Architecture

Figure 1 illustrates the technical architecture that we deployed to allow remote access to the Online-MQTT lab. The main logic of the Online-MQTT lab is implemented as a Virtual Machine (VM). The entry point of the user to our lab is the Moodle course of DigiLab4U, which, upon successful authentication, forwards the user to the web app instance running in the VM. This web app allows the user to control the experiment as well as receive real-time feedback through the camera live stream.

In order to ensure that only authorized participants access the remote lab, we use the open-source Learning Tools Integration (LTI) middleware

(IMS Global Learning Consortium, 2021), sharing a cookie for each user's web session of the experiment. Both LTI and WebApp instances (LabMS 1..n) run within Docker containers. In addition, we isolated the whole lab environment in its own Demilitarized Zone (DMZ).

The external DMZ defines the communication between the public Internet and Landshut university networks. The internal DMZ defines the communication between the virtual machine network (eth0) and the university network. Within the VM, we deploy scalable docker containers for each necessary module. Each hosted experiment runs in its own LabMS container and manages two robots (one RT and one non-RT) and one camera (eth1 and access point). The robots are restricted to communication with MQTT brokers only, whereas the camera streams are forwarded via WebRTC to the user's web session.

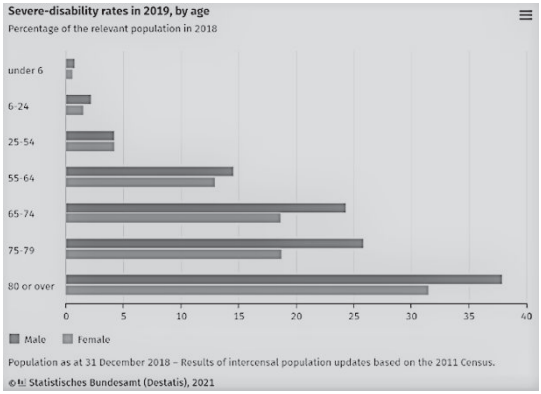

Figure 1. Hardware and Software Architecture of Online-MQTT

3.2 Integration with the DigiLab4U Infrastructure

There are several ways to implement remote lab access such as that Online-MQTT.

The open DigiLab4U (The DigiLab4U Project, 2021a) is one way that already provides for accounting, scheduling, booking, and billing using a very convenient interface. In addition, there is the virtual lab example code (LabMS), which offers a suitable template to start using DigiLab4U. Accordingly, DigiLab4U seems to be a unique opportunity for our Online-MQTT remote lab.

Using the DigiLab4U interface allows us to mainly focus on the implementation of lab-specific aspects. In detail, we do not have to handle (1) Moodle (Moodle, 2021), (2) its user registration and administration, (3) accounting incl. booking and scheduling, (4) billing, and (5) lab session handling.

The open and established LTI middleware manages the authorization of DigiLab4U Moodle-User to access our remote lab. This allows any authorized user to access our proposed lab at any time without implementing their own user management. Next, the RESTful LabMS software presents a plug & play method to manage our virtual lab sessions. The live feedback needed for users during their experiments can be easily realized by using this RESTful interface. No further session management is required. Also, scalability regarding multiple simultaneous virtual labs is easily provided by design. A simple configuration will add each virtual lab session to the open DigiLab4U infrastructure.

3.3 Enabling MQTT for LEGO EV3 Robots

We use robots that are based on LEGO MINDSTORMS EV3 due to their low cost and their ability to be easily rebuilt to simulate a wide range of machines in IIoT. Unfortunately, there is no MQTT support yet from LEGO. In addition, the EV3 processing and storage capabilities are very limited. Raspberry Pi, on the other hand, is an established multi-purpose, low-cost platform that supports MQTT well. Since there is an adapter to bridge EV3 and Raspberry Pi, i.e., the PiStorms, we decided to go for the following composition: robots are controlled by a Raspberry Pi via a PiStorms controller. The Raspberry Pi hosts an MQTT client and is connected to an open-source established MQTT broker on our VM, i.e., Mosquitto (Eclipse Foundation, 2021a). Publishers are then able to send commands to the robot. The main components of one robot are:

- EV3 LEGO parts used to build a 4-wheeled vehicle with two servomotors to drive the rear axle (actuators) and one obstacle detector (sensor).
- One Raspberry Pi (model 3B) implements the MQTT subscriber and controls the robot's movement through a python script depending on received MQTT messages. To this end, the Paho MQTT library (Eclipse Foundation, 2021b) is required. The Raspberry Pi connects to the Mosquitto MQTT broker via WiFi
- The PiStorms controller is the intermediate component between the LEGO platform and the Raspberry Pi platform, translating Python commands into EV3 commands.

A command to a robot is an MQTT published message, upon whose reception the robot moves a certain distance straight forward, referred to as one step.

3.4 Live Loop for Immersive User Interaction

From a User Experience (UX) perspective, it is essential to provide a live feeling to the user, where possible. Obviously, an interface that can be used to interact in real-time with the experiment is fundamental. In addition, a camera live streaming of what happens during the experiment is a powerful method with which to enhance the UX. In the following, we present our approach to building real-time interaction with our experiment while fulfilling the requirements above.

3.4.1 Enabling Web-based Interaction with the Experiment

The user interacts with the proposed lab through a web app dashboard implemented in Java, HTML, CSS, JavaScript, and Bootstrap (Figure 2). In order to start an experiment, the user specifies the values of the following three parameters:

1. *Number of Steps* each robot is supposed to move forward; the value should be $\in [1, 10]$ steps.
2. *Time Between Steps in ms*; value should be $\in [500, 2000]$ ms.
3. *Deadline in ms*—the latest time by which the message should be received—(Subscriber Reception Timestamp—Publish Timestamp) < Deadline; value should be $\in [1, 2000]$ ms.}

The number of steps is equal to the number of MQTT published messages that are going to be transmitted to each robot via a pre-specified topic on the broker. The user may publish the command to a robot instructing it to move the specified number of steps by clicking the button "start experiment" (Figure 2). After completion of the experiment, the student can visually observe the results in a plot.

3.4.2 Enabling GDPR-Compliant and Real-Time Control Feedback

At first glance, a camera seems to be the best fit for lively interaction between a remote user and the physical lab. However, camera deployment is subject to stringent data protection rules and such deployment should be compliant with the European General Data Protection Regulation (GDPR). To keep privacy protected, we decided to deploy the robots in a closet so that the camera's view was restricted and did not capture any human inside

the lab. The user could observe only the robots with the camera live stream. In addition, no microphone was used along with the camera.

For an emergent UX, it is essential to implement a streaming solution performing with low latency. Low-cost LAN accessible cameras are hard to configure for low latency streaming, as they usually make access to the stream only available through proprietary software (e.g. Smartphone App) and do not support open protocols such as the Real-time Transport Protocol (RTP) or Real-Time Streaming Protocol (RTSP). Other low-cost cameras offer a cloud-based stream, which unfortunately shows high or fluctuating latency, which massively degrades the quality of the UX. The Raspberry Pi Camera module is a low-cost but customizable alternative. Thanks to the open-source multimedia framework GStreamer (GStreamer Team, 2021), we succeeded in implementing real-time streaming using Web Real-Time Communication (WebRTC) from a Raspberry Pi camera. To this end, we deployed an instance of the open-source Janus WebRTC Server (Meetecho s.r.l, 2021). Access to Janus was secured via Stored Tokens on the Janus server.

A Token is interchanged every 24 hours. A Cronjob sends an API request to Janus to delete old tokens and generate new ones. To establish WebRTC connections, a Session Traversal Utilities for Network Address Translation (STUN) server has been integrated. In order to further enhance privacy protection, we connected the robots and cameras to the server via a dedicated and isolated WiFi Access Point, which is only reachable inside our experiment VM and has no connection to the Internet or to the campus network.

3.5 Considerations for Parallel Access to the Lab

We are providing access to shared physical resources with movement behaviour that requires mutual exclusion and prevents simultaneous access to a robot. Basically, an experiment can start only if both robots involved in the experiment are in a starting position. Given this hard restriction and in order to still accommodate multiple users simultaneously, we implemented two replication strategies, i.e., one in space and one in time.

We allow for many experiments that are independent of each other. Currently, we offer two experiments; however, due to our modular design and the support of LabMS, it is easy to scale to any number of experiments on demand (Redundancy in Space).

We keep experiments short and reset them automatically shortly after they are completed. Users that start a running experiment are informed from LabMS that the experiment is running and cannot be started now (Re-

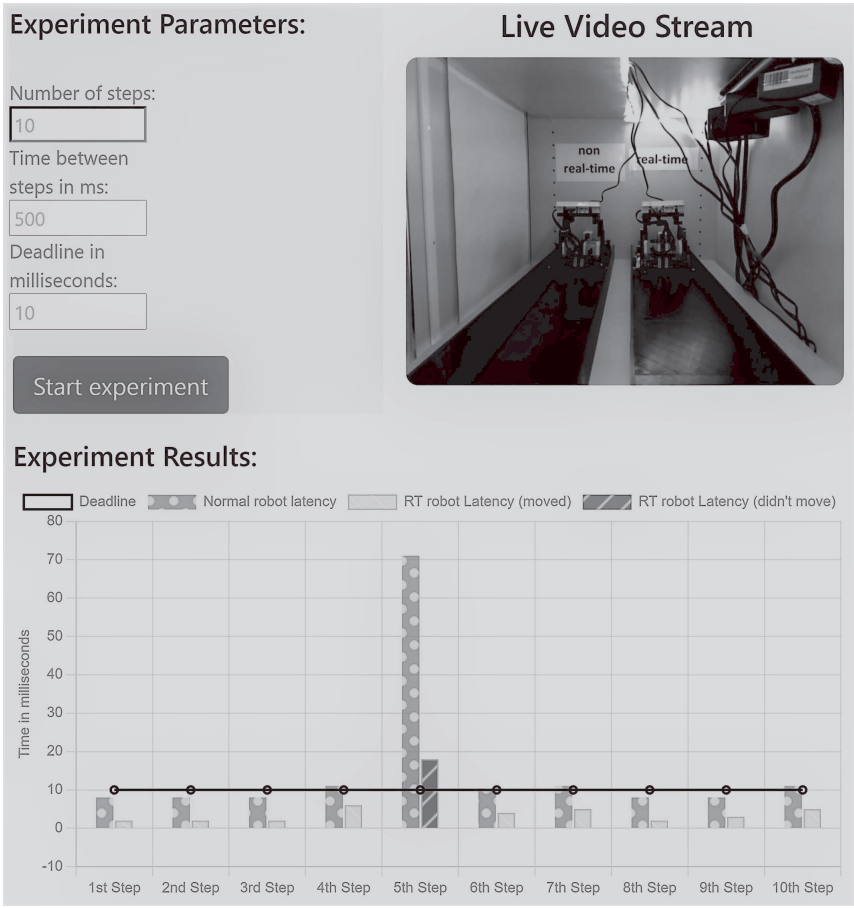

Figure 2. Control Dashboard for the Experiment

dundancy in Time). A shared calendar/scheduler in Moodle allows slots to be reserved to run the experiment. In addition, we isolated the experiments from each other in order to avoid failure propagation across experiments. Currently, we implement the isolation through software virtualization and MQTT settings, such as different topics for different experiments.

4 Evaluation

Now, we briefly revisit our requirements and assess to what extent our proposed Online-MQTT lab meets them.

Being web-based, all user interfaces, i.e., DigiLab4U and our lab interfaces, allow access from anywhere and anytime (24/7). In addition, for better scheduling of the physical resources, the DigiLab4U scheduler in Moodle allows convenient time slots to be reserved.

Through its modularity the proposed lab architecture containerization allows our experiment to be cloned within some hours. Given the two robots built as detailed in Section 3.3, the effort required to offer a new instance of the MQTT experiment is easily manageable, i.e., to create, deploy, and configure a new LabMS docker image, to configure LTI with shared keys, and add a web link to the new experiment in the Moodle course. Our software architecture is containerized where possible, allowing it to be scaled easily with regard to software. Being low-cost supports the scalability potential of the Online-MQTT lab. This highlights the scalability potential of Online-MQTT depending on user demands.

We built Online-MQTT using only low-cost, COTS components in order to easily scale it. This is mainly achieved through the selection of open-source hardware and software components such as Raspberry Pi, LEGO EV3, Docker, Mosquitto, Paho, etc. Being established with a rich and active ecosystem, the selected components require reasonable maintenance efforts. In addition, the components have a socket power supply, so that no batteries fail or need to be replaced. We decided to go for multi-purpose hardware platforms and architectures (Raspberry Pi and PCs) to run all elements of the MQTT protocol, i.e., clients and brokers. This allows us to overcome dependencies on the "machines", which then just need to have a bridge to a Raspberry Pi or a PC in order to be integrated into our lab. The containerization of the software simplifies efficient interchange of the components deployed and their extension. These design choices significantly enhance the extensibility of the Online-MQTT lab.

Through physical isolation of the online lab (its own closet, observed by cameras, and physical access for authorized personal only) and its cyber isolation (implementation of all the required cybersecurity countermeasures for opening resources such as DMZ and use of resilient open software like LTI), we achieved an appropriate trustworthiness level with regard to security, privacy, and safety that is satisfactory worldwide.

5 Conclusion

We proposed a novel lab that offers participants a journey from basic features up to advanced features of MQTT to address current research gaps. The proposed simulation of an RT scenario with the capability of live feedback enriches the learning experience for a broad range of participants. In this project, the unique DigiLab4U infrastructure has offered valuable support in overcoming the complexity of opening physical labs to remote users and one step toward the vision of an MQTT lab as a service. Nevertheless, some technical lab-specific barriers had to be overcome, which we briefly outlined in this paper. Fortunately, these solutions may be easily adopted by other (I)IOT labs for their digitization.

The online lab realized represents the basis for building a legolized architecture, where different protocols such as Sparkplug (The Eclipse Sparkplug Working Group, 2021), OPC-UA (The Open Platform Communications (OPC) Foundation, 2021), and CoAP (The Internet Engineering Task Force (IETF), 2021) can be selected by the user before running the experiment. This would allow for easy comparison and technology selection for the community. In addition, our lab architecture is open and safe so that WiP protocols can be tested and offered to the community. In particular, we will keep updating the current lab according to our future research findings so that students are able to try different RT classes and various approaches for automated and spontaneous subscriptions.

References

Eclipse Foundation. (2021a). Mosquitto [Online; accessed 28 Nov 2021]. mosquitto.org

Eclipse Foundation. (2021b). Paho: MQTT Python Client [Online; accessed 3 Dec 2021]. https://www.eclipse.org/paho/

French Ministry of Higher Education. (2021). FIT: An Open Large-scale Testing Infrastructure for Systems and Applications on Wireless and Sensor Communications [Online; accessed 29 Nov 2021]. https://www.iot-lab.info

Gambit Communications. (2021). The MIMIC MQTT Simulator [Online; accessed 29 Nov 2021]. https://www.gambitcomm.com/site/mqttsimulator.php

GStreamer Team. (2021). GStreamer — Open Source Multimedia Framework [Online; accessed 29 Nov 2021]. https://gstreamer.freedesktop.org/

HiveMQ. (2021). HiveMQ MQTT Broker [Online; accessed 28 Nov 2021]. www.hivemq.com

IMS Global Learning Consortium. (2021). Learning Tools Interoperability (LTI) [Online; accessed 30 Nov 2021]. http://www.imsglobal.org/activity/learning-tools-interoperability

Manzoni, P. (2021). An hands-on introduction to MQTT [Online; accessed 29 Nov 2021]. https://hackmd.io/@pmanzoni/BJ9hwSfhG?type=view

Meetecho s.r.l. (2021). Janus: the general purpose WebRTC server [Online; accessed 4 Dec 2021]. https://janus.conf.meetecho.com/

Moodle. (2021). Online Learning Management System [Online; accessed 2 Dec 2021]. https://moodle.org

Oasis. (2021a). MQTT Spec 3.1.1 [Online; accessed 19.11.2021]. https://docs.osis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html

Oasis. (2021b). MQTT Spec 5.0 [Online; accessed 19.11.2021]. https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

The DigiLab4U Project. (2021a). The DigiLab4U Mission [Online; accessed 29 Nov 2021]. https://digilab4u.com/mission-2/

The DigiLab4U Project. (2021b). The DigiLab4U Partner Labs [Online; accessed 29 Nov 2021]. https://digilab4u.com/labs/

The Eclipse Sparkplug Working Group. (2021). Mqtt + sparkplug = 'plug & play' iiot [Online; accessed 28 Nov 2021]. https://sparkplug.eclipse.org/

The Internet Engineering Task Force (IETF). (2021). The Constrained Application Protocol (CoAP) [Online; accessed 3 Dec 2021]. https://datatracker.ietf.org/doc/html/rfc7252

The Open Platform Communications (OPC) Foundation. (2021). OPC Unified Architecture (OPC UA) [Online; accessed 3 Dec 2021]. https://opcfoundation.org/about/opc-technologies/opc-ua/

Authors

Matas Führer
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
s-mjurev@haw-landshut.de

Roland Heinrich
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
s-rheinr@haw-landshut.de

Abdelwadoud Mabrouk
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
mabroukabdelwadoud@gmail.com

Tobias Christian Piller
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
tobias.piller@haw-landshut.de

Prof. Dr. Abdelmajid Khelil
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
khelil@haw-landshut.de

Kubilay Yildiz
Landshut University of Applied Sciences
Am Lurzenhof 1,
DE/84036 Landshut
www.haw-landshut.de
kubi.y@hotmail.de