Practical Education in IoT through Collaborative Work on Open-Source Projects with Industry and Entrepreneurial Organizations

Vladimir Podolskiy, Yesika Ramirez, Atakan Yenel, Shumail Mohyuddin, Hakan Uyumaz, Ali Naci Uysal, Moawiah Assali, Sergei Drugalev, Michael Gerndt
Technical University of Munich
Garching (near Munich), Germany
{v.podolskiy, yesika.ramirez, atakan.yenel, shumail.mohyuddin, hakan.uyumaz, ali.uysal, muawiya.assali, sergei.drugalev}@tum.de, gerndt@in.tum.de

Matthias Friessnig
UnternehmerTUM MakerSpace
Garching (near Munich), Germany
matthias.friessnig@maker-space.de

Anastasia Myasnichenko
UnternehmerTUM
Garching (near Munich), Germany
myasnichenko@unternehmertum.de

***Abstract*—This Innovative Practice Full Paper is devoted to the practices in the software engineering education that involve industrial and entrepreneurial partners. The particular focus of the paper is on the preparation, conduction and evaluation of the master practical course in the Internet of Things (IoT) open-source middleware. The motivation for the course is to train software engineers and architects that are able to solve the practical IoT challenges. The course was prepared and conducted during the winter term 2017-2018 at the Technical University of Munich. This course was created as a joint effort of the Chair for Computer Architecture and Parallel Systems, the manufacturing workshop UnternehmerTUM MakerSpace, and the entrepreneurship center UnternehmerTUM associated with Technical University of Munich. 14 participants of the course from different countries have developed an open-source IoT middleware, namely IoT Platform, that enables the retrieval of the data from sensors, distributed data storing and streaming, and platform services automatic detection. The middleware also incorporates well-documented application program interface (API). Performance monitoring facilities prepared by the team of students enable the Platform testing under different load. The IoT Platform was developed based on the 3D-printer management use-case from UnternehmerTUM MakerSpace and was presented at the UnternehmerTUM event Tech Challenge in a pitch format. Main contributions of the paper are in three areas: the methodology and technique to organize, conduct and evaluate the practical course in IoT that involves industrial and entrepreneurial partners; the pointers at particular technologies, solutions, and interconnections thereof for the development of new practically-oriented IoT courses; open-source IoT middleware that could be used for teaching and research purposes.**

***Index Terms*—Practical course in IoT, IoT Platform, IoT middleware**

I. Introduction

IoT has emerged in an attempt to capture, analyze and automatically manage the physical world in all its complexity. A use-case for IoT includes the sensors that collect data, a software that stores and processes that data, and actuators that manage the physical object. An example could include vibration sensors installed on a car motor with the software to analyze the collected data and to detect the possible motor wear. These data might be used for invoking the breaks in case of an immediate danger. Such cyber-physical systems, commonly known as automatic process control systems, were employed for a long time by the industry.

The IoT-enabling technologies became widely accessible: single-board computers with connected sensors sustain moderate price level, various open-source IoT software systems were developed. Industry envisions IoT as a mean to optimize its production and maintenance costs and to better engage the customer [2]. Industrial IoT (IIoT) makes the production sustainable and flexible, allowing to customize the goods [15].

Aiming to decrease the production costs with no negative impact on the product quality, industry refers to IIoT. The simplest IoT pipeline can be used to solve the majority of tasks related to these ultimate goals of any business [12]. Predictive maintenance is a clear example of a positive impact that IIoT may grant the industry [11]. Continuously collecting the data on the state of manufacturing equipment, a company can detect the piece of equipment that is close to a breakdown. By collecting and analyzing the data on the state of the produced goods, the company might be able to schedule the maintenance to avoid the product breakdown [10], [16]. Thus, IIoT allows manufacturing companies to improve the production process while reducing the associated costs.

To cover a variety of use cases, a company needs a general-purpose middleware able to adapt to a particular application. Such a middleware needs to implement the general data collection and processing pipeline. The application, in turn, might be depicted as a consumer connected to the application program interface (API). Such middleware is known under the umbrella term of IoT platform, e.g. Kaa IoT, Californium, Thingsboard.io, ThingSpeak, Geeny, GE Predix, SAP Leonardo, IBM Bluemix. Cloud services providers also introduce their cloud-native IoT services, e.g. AWS IoT, Google Cloud IoT, and Azure IoT Suite. As IoT market continues to grow, appears

978-1-5386-1174-6/18/$31.00 ©2018 IEEE

the challenge of getting the experts that can integrate various technologies in a single IoT solution [14], [18].

As a response to this challenge, the Chair for Computer Architecture and Parallel Systems (Chair) at the Technical University of Munich has established a master practical course with the goal to prepare the IoT middleware architects. As an outcome, each student should be able to design the comprehensive middleware solutions for IIoT and to work in a diverse team. The main deliverable of the course is the *IoT Platform*. The partners of the course MakerSpace and UnternehmerTUM have provided the real world IoT use case through the Tech Challenge: one of the MakerSpace teams at Tech Challenge has cooperated with the participants of the practical course in order to establish the support for their use case in the IoT platform. This collaboration has taken the shape of the matrix structure which resembles the organizational structure widely adopted in the industry.

In the following section we present the methodology used to organize and conduct the course. The third section covers the evaluation of the course efficacy using multiple evaluation techniques. The fourth section introduces the architecture of the developed IoT platform and the technologies used to provide the additional learning outcomes evaluation context and point at related technologies to support provision of the IoT-centered courses. The fifth section provides the basic performance evaluation for the developed solution allowing to assess the acquired competences of the students as software architects. The sixth section is devoted to the related works in the area of open-source IoT middleware. The final section encompasses conclusions and future work directions.

II. Master Practical Course in IoT Middleware Development

A. Course Goal and the Partners

The course evolved as a joint initiative of the Technical University of Munich, UnternehmerTUM, and UnternehmerTUM MakerSpace. The goal of the partners was to develop an IoT middleware to acquire the data from sensors installed on the manufacturing equipment in UnternehmerTUM MakerSpace, store it, and provide controlled access via API.

UnternehmerTUM MakerSpace is a 1,500 square meter high-tech facility on the campus of Technical University of Munich. MakerSpace provides users with access to machines, tools and software, as well as a community of creators. It offers a place to implement the ideas and innovative products in the form of prototypes and small batch production. MakerSpace provides various work areas such as metal and wood workshops, as well as textile and electronics facilities. In addition, 3D printers, laser cutters and a water jet cutter make it possible to form almost any shape.

MakerSpace hosts around 1000 users of manufacturing facilities and is a home of Technical University of Munich tech start-ups teams. The vast number of people and the size of the workshop render the use of an IoT-based system essential to track locations and conditions of tools, availability of consumables. Accumulation and use of these data would result in better user experience.

MakerSpace aims to extend the present one-way user-to-machine communication flow. A smart sensor system connected to the Internet and installed at all machines will gather machine-related as well as environmental data. Such a system enables the predictive analytics as well as the automatic adjustment of machines based on their black-box models derived using the machine learning techniques. The next reasonable step is the connection of the IoT gateway to the machines controllers. The role of the data storing, processing and provisioning component is imposed on the cloud-based IoT Platform developed by the students of the practical course. By implementing additional modules to the Platform, machine users can be informed about specific events via notifications messages or visualization charts. Such extensions could also support the intervention of the machine users into ongoing production processes.

Another course partner, **UnternehmerTUM**, has taken a role of the collaboration initiator by approaching the Chair with the idea of this course. From their side, UnternehmerTUM brought an entrepreneurial context allowing the participants to get a real IoT use case devised by interdisciplinary student teams at Tech Challenge and to pitch about the IoT Platform on the Tech Challenge Demo Day.

Tech Challenge is shaped as a 3-month long course with a strong focus on technology prototyping and business planning. During this course, students form interdisciplinary teams prototype their business ideas in the area of IoT, machine learning, artificial intelligence or robotics following the expectations of the partner companies such as Texas Instruments, MakerSpace paired with SAP and Munich-based start-ups Okiko and VUI.agency supported by UnternehmerTUM. Each partner formed own track with several student teams working on thematically close high-tech business challenges.

Tech Challenge is registered as a Technical University of Munich course with up to 6 ECTS points. The participating students get the prototyping budget (per team) and free access to machinery courses at MakerSpace, as well as coaching sessions in technology, team building, and business planning. Partners' representatives are reachable for the students for the whole duration of Tech Challenge. At the end of Tech Challenge, teams should go through a closed-door exam session. Afterwards, selected teams pitch their business idea and present the prototype on the Demo Day in front of an audience including partners, organizers, and their peers. Best teams are awarded with valuable prizes and receive support to convert their results into startup company.

B. Preparing the Course

The first preparation step was the clarification of the IoT middleware functionality using in-depth interview with the stakeholders. Throughout a series of discussions with the representatives of UnternehmerTUM a list of IoT Platform's functions was determined and a three-layered architecture of the system was proposed to support these functions. The performance evaluation framework for the platform was introduced afterwards. The outcome of these interviews was the document summarizing the functional requirements.

The learning outcomes were identified as follows:

1) the ability to design the compound middleware solutions integrating hardware with software to provide a suitable back-end for IoT-related use-cases;
2) the ability to select the existing software components based on the required functionality and performance;
3) the knowledge of technologies, software, and programming languages necessary to implement the designed solution in the area of cloud computing and IoT;
4) the ability to work in a *diverse* team of peers with different cultural background and to communicate the relevant issues to other teams;
5) the ability to present the results of the work clearly and understandably for people not possessing sufficient technical background.

The course was structured in three modules:

1) **Development of own toy IoT Platform**: the goal was to introduce the students to the relevant technologies and guide them through the whole design-and-implement process. This module solves the problem of skills and knowledge inequality of the participants. The module was allocated for 3 weeks of individual work.
2) **Implementation of the IoT Platform's core functionality**: the goal was to receive the Platform's prototype with the minimal necessary functionality. The module was allocated for 5 weeks of team work.
3) **Improvement of the IoT Platform and implementation of the features required by the partners**: the goal was to align the Platform with the requirements of the partners, to implement features relevant for its practical use. The module was allocated for 4 weeks of team work.

The steps required to implement the toy IoT Platform were thoroughly documented in the manual. It was placed on the course page in the TUM Moodle system.

The general outline of the course including the goals, outcomes, and the set of requirements was published on the Chair's website. The course program was additionally published on the TUMonline web portal used to enroll into the courses. Following, a preliminary meeting for the students willing to participate in the course was conducted.

The application procedure required a student to select the desired practical courses in the matching system of Technical University of Munich[1] and assign them priority (1 to 3, with 3 being the lowest) and to submit the supplementary materials to the course organizers (e.g. CV, transcript of grades, motivational email). When the registration via the matching system ends, course organizers give the priorities to the applicants (multiple levels, several applicants might be prioritized with the same level). After the matching is conducted, the course organizers receive the list of participants.

C. Conducting the Course

The major course workload was not bounded to the particular day or place. The only fixed part was the meeting conducted once a week at Technical University of Munich to discuss the progress in designing and implementing the platform. The ordinary meetings were not obligatory for participation, though the majority of the students participated in them to agree on data formats and convince other teams to add some data transfer functionality. One of the meetings was substituted by the guided tour at MakerSpace with the introduction of the manufacturing equipment and relevant IoT tasks.

Each module of the course ended with the presentation of the results. For the first module, students were randomly selected to present the toy IoT pipeline that they have implemented to the tutor. The second and the third modules were allocated for teams presentations and demos with the midterm presentation (second module) not including external experts. On the final meeting (third module) a demonstration of the Platform's functionality on the example of temperature sensors installed in 3D printers at MakerSpace was conducted. The demo included use cases of on-the-fly sensors connection, data retrieval, storing and access. The experts from MakerSpace and UnternehmerTUM gave their feedback to the participants.

The deliverables at the end of the course included the source code of the IoT Platform, the deployment scripts, the presentations explaining the architecture and the functionality of the Platform, and the documentation describing the activities to deploy, use, and modify the Platform. These deliverables were open-sourced on the GitLab of the Technical University of Munich.

One of the cornerstones of the practical course was its team work orientation. The assignment into teams was conducted based on the individual preferences of each student and the stack of technologies that student mastered. Messenger Slack was used as a mean of communication. Although work was organized differently in each team, a coordinating person was present in each group. The tutor intervention in the teams inner affairs was limited only to the situations of the emerging conflict, though such situations were rare (1-2 during the whole course).

III. Evaluating the Course Efficacy

A. Methodology

Due to the complexity and the interplay of the learning outcomes, the complementary course efficacy evaluation strategies were used [1], [3], [5]: project-based evaluation (Project), presentation-based evaluation (Presentation), and the evaluation based on written reports (Report). The main idea behind the selected evaluation procedure is that the success of the students at achieving the designed learning outcomes points at the efficacy of the course.

[1] Technical University of Munich employs the web system solving the stable matching problem with the algorithm by Gale and Shapley to assign students to the practical courses based both on their preferences and on the preferences of the course organizers.

The project-based evaluation assumes provision of the product which is a result of the joint work both within the team and between the teams [8]. The amount of work for a single team was initially designed in a way that it was not feasible to cope with it individually within the allocated one month of full-time work. By distributing the development of the components of the single system between different teams, an additional evaluation point was added - the successful implementation of the whole system heavily relied on the ability of each team to communicate with other teams. 3 out of 4 teams consisted of representatives of the different cultural background. As one of the teams was homogeneous, the mentioned workload distribution ensured that the participants of this team will be incorporated into the intercultural communication context. The readiness of the whole product allowed the implicit evaluation of the intercultural communication efficacy.

The demo- and presentations-based evaluation enables the assessment of the ability to present the results of work in a clear and structured manner. This strategy also contributes to resolving the challenge of the subjective assessment by involving independent domain experts in evaluating the presentations. This approach validated the learning outcomes evaluation acquired through project-based evaluation. Two presentations were conducted by the course participants as a part of this strategy.

The first presentation was conducted by one of the course participants at the final day of the Tech Challenge. Its major focus was on the idea and the architecture of the IoT Platform. The attendees of the Tech Challenge comprised experts from various industries (Texas Instruments, SAP, Okiko and VUI.agency), representatives of the UnternehmerTUM and MakerSpace, Technical University of Munich representatives (including course tutor), other course participants and other guests of the Tech Challenge Demo Day. The presentation was in a form of a 5 minutes-long pitch. An opportunity to ask questions after the presentation in a face-to-face form was given to everyone. This presentation aimed to receive the third party evaluation of the architecture and the selection of the technologies made by course participants. This added an external unbiased practice-backed evaluation.

At the second presentation the participants were required:

- to present the design and implementation of the Platform in a way understandable for non-experts;
- to demonstrate the functionality of the Platform in an interactive manner on a real-world use case;
- to cover significant implementation details for each Platform layer in distinct team presentation;
- to reflect on their personal experience participating in the course with particular emphasis on challenges.

The evaluating committee included the professor, representatives of UnternehmerTUM and MakerSpace, and the tutor. All the evaluating members were involved in the demonstration of the IoT Platform. They have asked questions about the functionality and provided their feedback for future extensions.

TABLE I
INDICATORS OF ACHIEVEMENT OF THE PLANNED LEARNING OUTCOMES

Outcome	Strategy	Achievement Indicator
1	Project	Functionality of each layer and the whole solution are successfully implemented and the code is provided
	Presentation	Capabilities of the solution are demonstrated during the presentation interactively
	Report	Clear textual description of the solution architecture with diagrams
2	Project	The performance of the implemented prototype is affordable for batch analysis applications
	Presentation	The concise motivation for the technologies selection is given during the presentation; the usability does not suffer
	Report	The clear motivation for the technologies selection and comparison results are provided in the report
3	Project	The provided solution code is clear and carefully documented
	Presentation	The questions about technologies used are answered precisely, no software faults occur during the interactive demonstration
	Report	Clear and repeatable solution deployment instructions are provided in the report
4	Project	Functionality of each layer and the whole solution are successfully implemented; code commits originate from different team members
	Presentation	All the teams are involved in the final demonstration of the solution; the presentations provide deep technical details on each part of students work
	Report	The provided report has in-depth technical details and at the same time homogeneous structure
5	Project	*Outcome not evaluated by this strategy*
	Presentation	Focused questions and related comments from the non-technical audience
	Report	Correct summary provided by the non-technical audience after reading the documentation

The evaluation based on the written reports included assessment of the reports independently provided by each team. Report should have included the description of the layer architecture, the actions required to deploy and use it, the motivation behind each architectural choice, and the reflection on experience of working within a team and collaborating with other teams. These reports allowed to evaluate the attainment of the software architectural thinking and ability to present the results of the team work in the clear and understandable way.

Table I matches the learning outcomes with the achievement indicators for each strategy.

A pair of **in-depth interviews (IDI)** was also conducted with the course participants 3 months after the grading. The goal was to zoom in into the individual experience and evaluate the achievement of outcomes at a finer level of granularity [5].

The next subsection provides the learning outcomes evaluation and IDIs highlights.

B. Evaluation

1) Learning Outcomes Achievement Evaluation: The evaluation was conducted using the indicators of Table I.

Outcome 1: The functionality of the whole solution was proved by students both during the practical course ordinary meetings and during the final interactive demo. The demo included the use case (provided by the Tech Challenge) of 3D-printer temperature measuring to identify the state of the machine (used/not used); the 3D-printer used was situated at the MakerSpace. The evaluation of the functionality by the tutor and external experts showed that all the required functions were implemented in the middleware. In addition to the requirements, a service discovery and a support for streaming API were added by students. The achievement of the indicators for this outcome points at the acquired ability of the course participants to design the *one-way* compound middleware IoT solutions for *data analysis applications*. The emphasized narrowed scope of the course is caused by the limited duration of the course. Therefore, the learning outcome does not cover such IoT middleware technologies as e.g. machine-to-machine communication and support for actuators.

Outcome 2: The acquisition of the ability to choose the software components of the IoT middleware was supported by the achievement of the corresponding indicators. The evaluation of the IoT Platform performance was provided by one of the teams. To ensure the independence of this team, the students were asked *to discover as many performance issues as possible for the future participants of the practical course to deal with*. The performance testing results demonstrated affordable performance; particularly, for the number of clients between 15 and 35 the maximum response time achieved 3 seconds which is too high for the real-time data analysis. In the presentations and in the reports, the participants have provided the motivation for specific technologies used: some technologies and tools were rejected due to the additional performance overhead (e.g. Apache Flink), whereas others were left aside because of their limited licences (e.g. InfluxDB, Elasticsearch X-Pack at the time of the course).

Outcome 3: Usually, the policy is to accept for the practical course only those students who possesses the relevant knowledge and experience. In the case of this course, the knowledge and experience in the relevant technologies was not fully covered by other courses. Thus, the relevant technologies were covered by a set of exercises that all the participants should have completed in order to proceed to the next stage. This added the necessity to evaluate the knowledge of technologies, software, and programming languages acquired at this stage. The achievement of this outcome was ensured by course participants providing the commented source code to the tutor and publishing it as an open-source solution thus taking the full responsibility for the quality of the code. The fast and comprehensive answers for the in-depth technical questions during and after the final presentations have shown the familiarity of the course participants with the building blocks of the produced solution.

Outcome 4: The direct evaluation of the ability to work in a diverse team of peers is not possible, though the overall performance of teams composed of the representatives with different cultural background can contribute to the evaluation of this learning outcome achievement. As was discussed in the evaluation of the *Outcome 1*, the functionality of the solution was fully implemented which points at the successful collaboration between the students both within a team and across different teams. As the participating master students have finished the bachelor programs in their home countries (based on the provided CVs), it should be valid to assume the lack of the intercultural communication prior to enrolling into the course. The participation of all the team members in the final presentations showed the involvement of each individual. Tracking the story of Gitlab commits demonstrated the involvement of each participant into the development process. The last point that contributes to the positive estimate of this outcome is that even given the opportunity to object against the inclusion in the specific team, no student asked to move her or him to other team with the representatives of the same cultural background.

Outcome 5: The evaluation of the students ability to present the results was majorly positive. After the presentations, experts from different areas have provided focused and relevant feedback about the possible use of the Platform and extensions. Further discussions with people of non-technical background, have revealed the overall understanding of the architecture and functionality, though the understanding of the hardware abstraction layer functionality was not precise - the mechanism of the server-backed data collection scripts deployment was not mentioned in discussions of the functionality despite being the major achievement. Therefore, the achievement of this learning outcome might be evaluated as sufficient as the additional room for improvement still exists.

Overall, all the learning outcomes were achieved by the students. However, it is necessary to emphasize the narrowed focus of the practical course - the scalability of the course scope may suffer with the inclusion of new technologies. This might result in an unaffordable load on the students. This problem might be approached by providing the preliminary lecture course introducing the students to the IoT-related technologies. This will also allow to remove the first course module aiming to ensure that all the participants have the common technological background.

2) In-depth Interview 1 Highlights:

On understanding the course goal and motivation: ”The goal of the course was to create a pipeline between IoT devices and end users. [...] My motivation was to learn more about IoT and communication between multiple devices via a central brain”.

On personal contribution: ”Service discovery. [...] I developed a solution by myself that mimics DNS [...] I still use this program for my cloud based projects”.

On background: ”[...] I hadnt heard about Kafka or Elasticsearch. I had also never used a Docker before, even though I heard about it”.

Personal challenges: ”Working in groups [...] you have to understand things you did completely so that you can share it with others. This creates a distinction between just copying, pasting code from the Internet”.

On work distribution: ”I convinced the group to use Kafka Streams so I knew that I had to deploy it myself. I was also working with some other to deploy the Kafka cluster to Docker. [...] If you proposed your group a different technology and convinced them to use it, you had to deploy it on your own [...]”.

On communication efficacy: ”[...] I was communicating very frequently with other people because our layer was a dependency to every other layer [...] There were more questions to ask and more topics to discuss with others as the project started to take shape”.

On development / project management balance: ”[...] Communicating with other members was not crucial for our layer but communication with us was very crucial to them, so most of our time was still spent on building the system”.

On acquired knowledge / experience: ”I experienced building a big project with more than 10 participants and with other interested parties [...] I also learned the cutting edge technologies in cloud software and explained technical topics to other people”.

On relevance for career: ”I learned that you have to be very skeptical about the first version of your project [...]”.

On what could have been done differently: ”I would compare and analyze proposed technologies better before going right into coding [...]”.

3) In-depth Interview 2 Highlights:

On motivation: ”Since cloud computing and IoT are trending topics in IT, I wanted to learn more about it [...] it was mentioned that we would have a collaboration with the MakerSpace which motivated me because the projects would have a real impact”.

On requirements: ”The requirements regarding system architecture and stack of technologies to use were flexible. Some suggestions about technologies were presented but we had the freedom to choose based on our expertise and research”.

On background: ”[...] Regarding technologies I was already comfortable with different programming languages, configuring virtual machines and using asynchronous communication tools for distribution of work in a team”.

Personal challenges: ”To make a choice regarding the technology to use for stream processing [...] Definition of common interfaces and communication between teams [...] In my team I was in charge of the configuration of Kafka focusing on ensuring the fault tolerance. As it was my first experience with this technology, it was challenging but also fun”.

On work distribution: ”At the beginning there was some confusion about the design of the implementation [...] there were some miscommunication issues between the team like a team member would end up spending time trying to implement some task that another member was already responsible for. It changed after discuss it in some weekly meetings. Also there was miscommunication with other teams, since sometimes the requirements were not clearly specified. We solved this misunderstandings mainly on the weekly meetings [...] From a list of tasks, each person selected the one that each would like to work on”.

On communication efficacy: ”[...] Depending on the assigned task I would communicate often (around 2-3 times per week) mainly with team members who had some relation to that specific task [...]”

On development / project management balance: ”Since we were a small group, communication within my team didn’t required too much time [...] for communication with other teams, there was a person in charge of it [...]”

On acquired knowledge / experience: ”I got basic knowledge about the whole pipeline [...] in addition I got a deeper knowledge about data processing, storing and streaming tools which was the part that I wanted to focus on”.

On relevance for career: ”The acquired knowledge about technologies used to store and process stream data [is the most important for the future career]. In addition the experience of making design decision”.

On what could have been done differently: ”I would still participate on the layer for data storage and processing but I’d expect a real world use case which would challenge me [...]”

C. Lessons Learned

The following directions for the improvement were identified at the end of the course:

- the involvement of the partners should be based on the continuous work of students on-site;
- use cases requirements should be accurately identified from the start;
- an additional technological background in IoT needs to be provided in the form of a lecture course;
- the tutor should invest more time into building a community of interested students around the project.

IV. IoT Platform

A. Motivation

Although it may seem controversial to provide the technical details of the IoT Platform implementation in the educational paper, the motivation behind this section is:

1) to help the reader understand how one of the outcomes was achieved by the participants of the course,
2) to suggest pointers at the technologies that might be valuable for the development of other IoT courses.

B. Overview of the Architecture

IoT Platform is composed of 4 loosely coupled layers that work as a solid solution.

The first layer (I) provides data streaming functionality for any sensor connected to one of the single-board computers (SBC) used. The data is compressed and sent to the layer II for the processing. The first layer has a built-in mechanism to restore the streaming in case of failures. The diversity of the IoT hardware is supported through the pre-configured integration of many common SBCs and sensors.

The second layer (II) receives measurements from the first layer, decompresses and stores them in a queue to retrieve later without duplication. Layer II delivers the data to the API layer both in real-time as a stream and in chunks as a response to a request. Large amount of data generated by sensors has enforced the implementation of a set of strategies to guarantee the scalability using an Apache Kafka cluster for processing and an Elasticsearch cluster for storing.

The third layer (III) supports the provision of either the historical data via the common REST API or the real-time measurements via the publish/subscribe interface. An authentication mechanism allows only the authorized users to access the data.

An additional functionality was implemented to monitor the performance of the platform. This functionality also includes the configurable testing component. The performance tests can be carried out individually on the layers II and III.

Figure 1 shows the information flow between layers. The following sections explain the implementation of each layer.

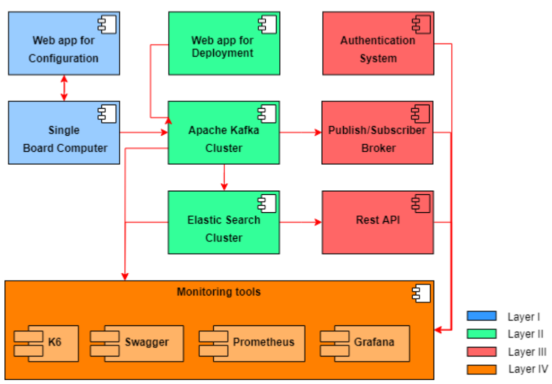

Fig. 1. Component Diagram of Technical University of Munich IoT Platform

C. Layer I: Flexible SBCs and Sensors Framework

Implementation of an extensible and flexible sensors framework that can host any type of sensor requires an abstraction of SBC with a sensor-independent architecture. The most common interfaces and widely-used communication protocols should be directly supported in the software. The diversity of hardware to support imposes the requirement to make the structure of the framework modular which entails the hierarchy of abstractions corresponding to various interfaces, sensors, etc. Each component of the layer should be extensible, since new types of SBCs and sensors appear frequently.

The layer supports **23** various sensors, including temperature, pressure, humidity, and distance sensors. Developers could easily add the support for new sensors by extending the input-output interfaces.

The system can be reconfigured to use different endpoints and different protocols for the sensor groups. Any new sensors that are integrated into the system during the prototyping process can be added to the configuration file by the user without any need to intervene with the underlying code. After the restart, the system will run with the new configuration.

To make the integration of hardware easier, a web application with its own user-interface (UI) was implemented. The web application allows users to register new SBCs, create, update and store their settings files permanently. It also allows user to examine SBCs log files. After restarting SBCs to reflect the changes in the configuration files, second component of the platform (start-up script) starts running. This script allows user to select respective SBC via command line interface (CLI) to fetch SBCs settings file from the web application and starts fetching data from configured hardware.

In order to send the data to the data storing and processing layer (Layer II), the communication manager creates the Kafka producers using IP-addresses, ports and topics specified in the configuration file. The JSON packets are sent with the board ID and the device ID for each collected measurement.

D. Layer II: Scalable Data Storing and Processing

Layer II of the platform is responsible for assembling sensors measurements into JSON strings, storing and processing them. The configuration of this layer ensures the reception of messages without duplication. The following components contribute to the most of the layer's functionality: a stream processing application, a cluster of Elasticsearch database instances, and a cluster of Apache Kafka instances.

1) Apache Kafka Cluster: Apache Kafka provides the messages streaming functionality for the layer. This layer's services are packed in Docker containers. Docker Compose is used to provide the redundant copies of processing units. Kafka requires Zookeeper, therefore the default configuration of the Platform encompasses 3 instances of Zookeeper and Kafka each, running in 6 different Docker containers.

2) Elastic Search Cluster: The Elasticsearch cluster of the Platform consists of a master node that is connected to slaves. The clustering supports scalability & availability of the data layer. Whenever one of the nodes goes down, the system will continue to function normally without losing the data. The configuration of shards and replicas contributes to achieving this goal. A shard holds a portion of data - documents with the relevant properties. Sharding implements the horizontal scaling for Elasticsearch. Replicas are considered as redundant copies of shards. They provide high availability of data in cases of a single node failure and increase the performance when carrying parallel search processes.

3) Stream Processing Application: Stream Processing Application is a Java-based component of the second layer that integrates Apache Kafka and Elasticsearch into single data messages storing and processing pipeline. The integration with Apache Kafka is made via the Java stream access libraries.

4) Service Discovery: Service discovery functionality was introduced as the connection between different layers of the platform requires the knowledge of IP-addresses for the virtual machines which could be reassigned after restarts. Service discovery mimics the Domain Name System (DNS) and stores the IP-addresses of all the VMs to a database that can be reached with a static hostname. A client program runs at every restart, sends the IP and the name of the host to the server, retrieves the IP-addresses of all the machines and updates the environment variables so that other programs can use them.

E. Layer III: Batch and Stream Access to the Sensors Data

IoT Platform supports two basic types of data access: streaming access in real time (publish-subscribe) and batch access to the historical data stored in Elasticsearch (pulling). Both access endpoints utilize same security module which provides user authentication and authorization. The communication channel is forced to use secure TLS protocol in all cases (HTTPS, MQTT with TLS) to ensure the data protection against man-in-the-middle attack.

Authentication and authorization employ the internal layer storage module (RQLite database). Along with the credentials, the database stores the list of sensor boards which are accessible to the specific user. For every connection the system checks if the user is authenticated and authorized to use the resource.

API server provides RESTful endpoints to retrieve historical sensor data from Elasticsearch. The server implementation is based on Node.js and Express.js. The data can be requested both on the level of sensor boards and on the level of particular sensors. The user may define the time frame for the data.

Stream data access is ensured by the MQTT broker component RabbitMQ. The component is powered by Kafka-MQTT module which serves as a "bridge" between the raw data input and easy-to-use output.

F. Performance Monitoring

The following software was used to monitor the IoT Platform performance:

- **K6** - an open source load testing tool.
- **InfluxDB** - an open source time series database.
- **Prometheus** - an open source time series database with a built-in monitoring functionality.
- **Grafana** - an open source dashboard and graph editor tool used to visualize the collected metrics.

1) Monitoring for Data Storing and Processing Layer: Combination of InfluxDB with Prometheus was used to collect the resource utilization data. Node exporter was used to measure and monitor the server side infrastructure including CPU, RAM, and I/O usage. Grafana provided visualization of metrics values collected by Prometheus on a dashboard.

2) Monitoring of the API Performance: The monitoring of the API layer is supported by the Swagger API Framework. An Express.js middleware swagger-stats is used to trace API calls and monitor individual calls as well as their health and usage statistics. Swagger-stats exports collected metrics in the Prometheus format, following a new scrape configuration.

V. Evaluation of the Platform Performance

The load testing included repeat of the API requests every second by all the virtual users.

The load test outline is as follows:

- Maximum Number of Virtual Users: 35.
- Duration of the test: 60 sec.
- Phase 1 (0-20 sec): 0-15 VUs.
- Phase 2 (20-40 sec): 15-35 VUs.
- Phase 3 (40-60 sec): 35-0 VUs.
- Maximum response time for Phase 1: $<$500 ms.
- Maximum response time for Phase 2: $>$3 sec.
- Average response time for Phase 1: $<$300 ms.
- Average response time for Phase 2: 1.54 sec.
- Layer 3 CPU usage at maximum load exceeded 100%.

Although the system withstood the load, Grafana showed that with the number of VUs approaching the maximum value of 30, the response time increased from less than 300 ms to 3 seconds which is a 10-fold increase. The results clearly point at the bottleneck in one of the layers. Further investigation showed that the bottleneck was introduced by the API layer deployment with a CPU usage of more than 100% (several CPUs) during the load test.

VI. Related Works

Claimsware middleware presented by George and Mahmoud addresses the security challenge introducing the notion of claim - a statement that distinguishes between different entities participating in the IoT use-case [7]. A middleware solution MinT provides comprehensive hardware abstraction level and allows for energy-efficient device and data management [9]. Hydra supports service disovery and enables the use of devices as services [6]. The processing of the data in Hydra rather than on the IoT devices could become a disadvantage for the real-time processing tasks. UbiSoAP is a lightweight service-oriented middleware solution [4]. It is designed for the integration with ubiquituos networks, but also supports heterogeneous networking devices as well as the legacy web services. MOSDEN enables sensing-as-a-service model and has a plugin-based architecture which enhances the use of the middleware in the presence of heterogeneous devices [13]. Xively offers middleware services for IoT solutions [17]. Even though this middleware supports multiple data format, it creates an overhead in the system since there is no way to homogenize the incoming data.

VII. Conclusion and Future Work

The paper highlights the organizational details and challenges of conducting the practical course in design and development of the IoT middleware in the collaboration with industrial partners and entrepreneurship organizations. The evaluation of the achievement of the learning outcomes showed the overall efficacy of the course.

The course could be modified in a number of ways. First of all, it could also be provided in the collaboration with electrical and mechanical engineering departments. Implementation of the support for the actuators could extend the scope of the course into the area of IIoT. Involvement of the additional industry partners could contribute to the diversity of the use cases and provide the focus at particular industry-relevant technologies. Last but not least, a preliminary lecture course in IoT-enabling technologies with exercises could reduce the scope of the practical course and allow students to focus on the software architecture and use cases integration tasks.

REFERENCES

[1] Moti Frank * and Abigail Barzilai. Integrating alternative assessment in a project-based learning course for pre-service science and technology teachers. *Assessment & Evaluation in Higher Education*, 29(1):41–61, 2004.

[2] H. P. Breivold. Internet-of-things and cloud computing for smart industry: A systematic mapping study. In *2017 5th International Conference on Enterprise Systems (ES)*, pages 299–304, Sept 2017.

[3] Bull J. Brown, G. and M. Pendlebury. *Assessing student learning in higher education*. London: Routledge, 1997.

[4] M. Caporuscio, P. G. Raverdy, and V. Issarny. ubisoap: A service-oriented middleware for ubiquitous networking. *IEEE Transactions on Services Computing*, 5(1):86–98, Jan 2012.

[5] Yaron Doppelt. Implementation and assessment of project-based learning in a flexible environment. *International Journal of Technology and Design Education*, 13(3):255–272, Oct 2003.

[6] M. Eisenhauer, P. Rosengren, and P. Antolin. A development platform for integrating wireless devices and sensors into ambient intelligence systems. In *2009 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*, pages 1–3, June 2009.

[7] V. M. George and Q. H. Mahmoud. Claimsware: A claims-based middleware for securing iot services. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 649–654, July 2017.

[8] Yasemin Glbahar and Hasan Tinmaz. Implementing project-based learning and e-portfolio assessment in an undergraduate course. *Journal of Research on Technology in Education*, 38(3):309–327, 2006.

[9] S. Jeon, C. Lee, Y. Han, D. Seo, and I. Jung. Mint: Middleware for coope rative interactions of things. In *2016 IEEE International Conference on Consumer Electronics (ICCE)*, pages 127–128, Jan 2016.

[10] D. Jung, Z. Zhang, and M. Winslett. Vibration analysis for iot enabled predictive maintenance. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 1271–1282, April 2017.

[11] A. Kanawaday and A. Sane. Machine learning for predictive maintenance of industrial machines using iot sensor data. In *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 87–90, Nov 2017.

[12] S. A. Nissanka, M. A. J. R. Senevirathna, and M. Dharmawardana. Iot based automatic storing and retrieval system. In *2016 Manufacturing Industrial Engineering Symposium (MIES)*, pages 1–5, Oct 2016.

[13] Charith Perera, Prem Prakash Jayaraman, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Mosden: An internet of things middleware for resource constrained mobile devices. In *Proceedings of the 2014 47th Hawaii International Conference on System Sciences*, HICSS '14, pages 1053–1062, Washington, DC, USA, 2014. IEEE Computer Society.

[14] J. R. Pimentel, O. Baltuano, R. Chan, and J. P. Tincopa. Teaching and learning engineering subjects in the times of the iot: A case study on a course on wireless communications and networks. In *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, pages 3589–3594, Oct 2017.

[15] F. Shrouf, J. Ordieres, and G. Miragliotta. Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm. In *2014 IEEE International Conference on Industrial Engineering and Engineering Management*, pages 697–701, Dec 2014.

[16] D. Shyamala, D. Swathi, J. L. Prasanna, and A. Ajitha. Iot platform for condition monitoring of industrial motors. In *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*, pages 260–265, Oct 2017.

[17] N. Sinha, K. E. Pujitha, and J. S. R. Alex. Xively based sensing and monitoring system for iot. In *2015 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6, Jan 2015.

[18] L. Zhaobin and L. Wenzhi. Exploration and consideration of practice teaching on iot comprehensive practice base. In *2017 12th International Conference on Computer Science and Education (ICCSE)*, pages 319–323, Aug 2017.