This WASA course unit [CM-W-INT] describes the content and the organization of the lecture and practical course "Web Applications and Service-oriented Architecture" (WASA) provided by the research group Cooperation & Management (C&M, Prof. Abeck). Current concepts of software development and architectures (including Behavior-Driven Development, Domain-Driven Design, Microservices, RESTful Webservices, 12 Factor App, CI/CD Build Pipelines, DevOps, Container-virtualized Infrastructures) as well as related standards and technologies (including HTTP, Java, JavaScript/TypeScript, Angular, Spring, GitLab-CI, Docker, Kubernetes, Prometheus) are introduced which are needed to develop advanced (i.e. microservice-based, IoT aware, cloud-native, mobile) web applications. The web applications stem from the connected car domain which includes concepts from the domain of Internet of Things. Since the concepts presented in the lecture must be practically applied to really understand them the WASA lecture is only offered in a combination with the WASA practical and/or seminar course.

IMPORTANT: In the summer semester 2021, the lecture and practical/seminar course are organized as an online event. This format is called WASAOnline.

The WASAOnline kickoff lecture will take place on Wednesday, 14th April 2021 at 10:00 am.

Each student who wants to take part in WASAOnline and in the kickoff lecture should send an email to Sebastian Abeck (sebastian.abeck@kit.edu) to receive the web link to the web session which is made available for his/her personal use. Please use your depseudonymized KIT student email address (see https://my.scc.kit.edu/shib/pseudonymisierung.php for further information). Thank you!

The lecture material is made available in English. During the online lecture, the content is presented and discussed in German. The oral examination is conducted exclusively in German. Each student can choose if he/she wants to write his/her practical/seminar thesis in German or in English. Thesis templates are made available in LaTeX.

C&M Cooperation & Management
KIT Karlsruhe Institute of Technology
WASA Web Applications and Service-oriented Architectures

The research work carried out by C&M can be divided into two main areas:

(Connected Car, Healthcare, Internet of Things) In this area connected car and healthcare application and application based on the concept of domain modeling and microservice architectures are developed. Relevant concepts applied in the development process include Behavior-Driven Development (BDD) and Domain-Driven Design (DDD). All microservice-based applications developed by C&M include aspects of Internet of Things (IoT). An important standard on which our implementations are based on is the SensorThings API from the Open Geospatial Consortium.

(DevOps, Identity and Access Management) DevOps concerns the continuous integration (CI) and continuous deployment (CD) of microservice-based software systems into a container-virtualized (Docker/Kubernetes-based) cloud infrastructure. Identity and Access Management (IAM) is a highly relevant crosscutting concern appearing in every web application. One of the projects is concerned with the provisioning of IAM-as-a-service (IAMaaS) provided in a cloud infrastructure.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDD</td>
<td>Behavior-Driven Development</td>
</tr>
<tr>
<td>CI/CD</td>
<td>Continuous Integration / Continuous Deployment</td>
</tr>
<tr>
<td>DDD</td>
<td>Domain-Driven Design</td>
</tr>
<tr>
<td>IAM</td>
<td>Identity and Access Management</td>
</tr>
<tr>
<td>IAMaaS</td>
<td>IAM-as-a-Service</td>
</tr>
<tr>
<td>IOSB</td>
<td>Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation (Optronik, Systemtechnik und Bildauswertung)</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>DevOps</td>
<td>Development and Operations</td>
</tr>
</tbody>
</table>
The software development process applied by C&M combines the concepts of Behavior-Driven Development (BDD) and Domain-Driven Design (DDD). Both concepts provide complementary contributions to the layered microservice architecture as the figure illustrates.

(1) An implemented feature can be seen as a deployable increment of the software system. (Feature 1, Feature 2, …) The ordering of the features in the figure implies that the first feature should cover the core functionality of the software system.

(2) The domain model makes sure that the static and dynamic domain knowledge is consistently used by each feature. This ensures that the features build a consistent whole although each feature is developed and deployed independently from other features.

BDD  Behavior-Driven Development

DDD  Domain-Driven Design
(PredictiveCarMaintenance, ClinicsAssetManagement, EnvironmentalMeasurementDashboard) These are the microservice-based applications which are developed by C&M in cooperation with his partner institutes.

(IAM Service, Auth0) The Gherkin features include Identity and Access Management (IAM) requirements, such as authentication of the user of the PCM application. The IAM service Auth0 is used to implement these requirements.

(PCM Microservice, The Twelve-Factor App) The application itself is based on the microservice architecture. The development of the PCM microservice should follow the Twelve-Factor App requirements resulting in a cloud-native application.

(Message Broker, RabbitMQ) The communication between the Backend-For-Frontend (BFF) and microservices is based on the exchange of events. The event bus is provided by a message broker tool named RabbitMQ.

(PCM Microservice, Connected Car API, OpenAPI, IoT, IoT Service, SensorThings, Web of Things) The information needed for predictive maintenance is provided by a connected car's API specified in the OpenAPI format. A connected car today is one of the most relevant "Things of the Internet". Therefore, IoT standards, such as SensorThings API from OGC and Web of Things from W3C, are taken into account in the development of the PCM application.

(API Security, Connected Car API) The API offered by the connected car must be protected against unauthorized access. The IAM service provides the authentication and authorization information needed to take the access decision at the API.

(DevOps, Build Pipelines, Kubernetes Cluster, Prometheus Monitoring) IC2020 will also make relevant contributions to the highly relevant DevOps topic. The goal is to develop a build pipeline based on GitLab-CI by which the PCM application is continuously integrated (CI) and deployed (CD) on a Kubernetes cluster. The PCM application provides metrics which are monitored using the tool Prometheus.

(IoT, SensorThings API, FROST) Things of the Internet offers Sensor Data which will be processed and stored into a database in the IoT domain. A possible implementation of the SensorThings API is the FROST-Server. The Sensor Data of each Thing can have its own Data Format. This SensorThings API data format is used for standardization.

(CAM) The ClinicsAssetManagement (CAM) controls medical devices in a healthcare context. One goal of the CAM is to offer indoor position data of the medical devices for the users. Therefore functionalities of the IoT domain will be used. In the Healthcare domain one of the most used data standards is Fast Healthcare Interoperability Resources (FHIR) which has the aim to enable the digital exchange of medical data between different systems. The CAM application uses the FHIR standard to manage the devices. One challenge is to enable the interoperability between the Healthcare and the IoT domain with their different standards.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCM</td>
<td>PredictiveCarMaintenance</td>
</tr>
<tr>
<td>CAM</td>
<td>ClinicsAssetManagement</td>
</tr>
<tr>
<td>FROST</td>
<td>Fraunhofer Open Source Sensor Things API</td>
</tr>
<tr>
<td>FHIR</td>
<td>Fast Healthcare Interoperability Resources</td>
</tr>
</tbody>
</table>
The table describes the planned sequence of course units presented during the lecture.

(Content) All titles printed in upper case are course units that are the basis for the oral examination carried out at the end of the semester. After the course unit was held a range of page numbers are added. This indicates which pages must be prepared by the student for the oral examination.

COURSE UNIT (P. X – P. Y) All course units written in upper case letter are primary examination material. If page ranges are indicated only these pages are treated in the oral exam. No page range means that questions of the complete course unit can be asked in the oral exam.
The acronym WASA stands for "Web Applications and Service-oriented Architectures". Four different types of WASA courses are offered: (i) lecture courses WASA1 and WASA2 (ii) practical courses WASA1 and WASA2 associated to the lecture courses (iii) prosemear course associated to WASA1 lecture course and seminar associated to WASA2 lecture course (iv) key qualification course (germ. Schlüsselqualifikation SQ).

(1) The lecture courses WASA1 and WASA2 each comprise 2 semester hours. A student who attends one of the lectures acquires 4 credit points (germ. Leistungspunkt).

(2) The practical course runs in parallel with the lecture course and counts 5 credit points meaning a workload of 150 hours. The capacity of students C&M can offer the practical course depends on the current projects carried out in the research group.

Hint: In the Wirtschaftsinformatik study programme the name of the module is "Microservice-basierte Web-Anwendungen".

(3) The examiners are Prof. Abeck and one of the C&M's Ph.D. Researchers. Since the examination is in the lecture term the students should have a good personal resource management in order to have enough time for the preparation of the examination.

SQ Schlüsselqualifikation (Key Qualification)
Characteristic of the Offered 
WASA Practical/Seminar Courses

(1) The practical/seminar work is carried out in a project team 
  (1) A student doing his/her practical work at C&M takes the role of a 
      JuniorStudent co-supervised by a SeniorStudent/PhDResearcher 
  (2) The team meets each week for about one hour 

(2) JuniorStudents work together with the co-coaching 
    SeniorStudent/PhDResearcher on specific topics that are relevant for his/her 
    Bachelor/Master/PhD Thesis 
  (1) JuniorStudent continuously provide contributions that are discussed in the 
      team (Continuous Writing CW) 
  (2) Reviews by the SeniorStudent/PhDResearcher provide helpful feedback 
      (Continuous Review CR) 
  (3) The work packages defined during the practical/seminar course depend on 
      the results and knowledge gained from the work done so far 
  (4) Own ideas how to deal with a topic are welcome 

On this page the specifics of the WASA practical/seminar courses which are offered in parallel to the 
WASA lecture are described.

(1) A project consists of about 4 to 6 practical/seminar students. 
(1.1) A SeniorStudent is a student who is writing his/her bachelor thesis or master thesis at C&M. 
(1.2) The meetings take place at a defined time which is fixed at the beginning of the semester.

(2) The topics dynamically evolve from the work done by the SeniorStudent in their bachelor/master 
thesis. 
(2.1) The JuniorStudent should actively participate in the discussion and make own proposals how the 
topic should be treated. 
(2.2) Reviews are an integral part of the work in the project team. 
(2.3) This means that the practical/seminar work has a dynamic characteristic. 
(2.4) There is a high flexibilty and liberty concerning the focal points of the practical/seminar work.
Working in a Project Team

(1) Each project team meeting is prepared by an agenda to which all team members provide
(1) Markdown document in GitLab
(2) Weekly project team meetings starting next week
(3) Project team members communicate in German

(2) JuniorStudents can write their practical/seminar thesis in
(1) German or English (preferred)
(2) LaTeX (Overleaf as the preferred tool)

(3) In the first week all project team members become acquainted with the C&M-TEAMARBEIT

The project team agenda is a markdown document by which the work of all project team members is coordinated.

(1) The project team leader takes care that all team members contribute to the agenda which is available for each project team meeting.
(1.1) For each project team a subgroup exists in the C&M GitLab. In the README file of the repository "Projektteamtreffen" the agendas of the project team meetings are collaboratively worked out.
(1.2) The project team meets for one hour every week. The day and time are defined by the project team members.
(1.3) That is why the agenda is worked out in German.

(2) There are templates available for the different variants in which a JuniorStudent can write his/her practical/seminar thesis.

(3) C&M-TEAMARBEIT is a document [CM-CMT] which describes how the members of the research team C&M efficiently work together.

(Screen dump on the right hand side) This is an excerpt of the GitLab document "Projektteamtreffen" which provides an agenda draft of the first two project team meetings.

Each project team makes a specific contribution to the overall microservice-based system environment.

(PT Schneider) ...

(PT HippchenSidler) ...

(PT Sänger) ...

(PT Throner) ...
The goal within this project team is to provide Internet of Things (IoT) aspects, such as the position data of a medical device to the ClinicsAssetManagement (CAM) application. When no standard is used, different data formats decrease the interoperability (since different devices have different data formats). This leads to the typical interoperability problem within IoT. Therefore, the sensor information is standardized using the SensorThings API from the Open Geospatial Consortium (OGC).

In order to separate IoT and medical data, a cooperation with PT MetinMüller is established.

In the following, the work packages of PT Schneider team are described.

The separation of the IoT and business concerns need to be expressed in the domain models of the healthcare and IoT domain. This task is done in cooperation with PT MetinMüller.

In order to provide the IoT data, the identified IoT services are modeled. The IoT extension could be shown by extending the application map. The extension of the application map shows how the IoT aspects are integrated into the CAM application. Afterwards, the services are implemented. In addition, the services are modeled using relation views.

The IoT information, such as sensor information needs to be mapped to the SensorThings API standard. Fraunhofer's FROST-Server is used as SensorThings API implementation. A FROST-Server instance already exists to fulfill this work.

The SensorThings API stores the IoT data and can be seen as the interface for accessing the IoT data. Therefore, the available FROST-Server instance may need some adjustments.

<table>
<thead>
<tr>
<th>CAM</th>
<th>ClinicsAssetManagement</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROST-Server</td>
<td>Fraunhofer Open Source SensorThings API Server</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>OGC</td>
<td>Open Geospatial Consortium</td>
</tr>
</tbody>
</table>
(1) The goal of this project team is the modeling of the existing Environmental Measurement Dashboard (EMD) application of the Landesanstalt für Umwelt Baden-Württemberg (LUBW) by using Domain-Driven Design (DDD). This is a necessary step as the current software architecture is not designed using modern approaches for modeling microservice architectures.

(2) One of the crucial steps in the software architecture building process is the identification of missing components. In the current state, some of the functionality is misplaced and consequently, domain and application logic must be separated, and more application functionality has to be encapsulated in their own components.

(3) The JuniorStudents work on the following work packages in the context of the project team.

(3.1) The existing application has to be examined and analyzed to get knowledge about its functionality and architecture. Within this process, documentation has to be created based on the previous analysis. Several artifacts (e.g. context map, API specification, …) are created which are based on the previous analysis. They contain information how microservices are designed and encapsulate functionality in the right place.

(3.2) Based on the previously created artifacts, a some of the microservice candidates are implemented. In this step, especially those components which potentially do not exist in the initial architecture are of high interest.

(3.3) In the last step, the microservice (candidates) are integrated into the C&M Developer Portal. This step is achieved in cooperation with PT Throner and/or PT SängerHanselmann.

API Application Programming Interfaces
DDD Domain-Driven Design
EMD Environmental Measurement Dashboard
LUBW Landesanstalt für Umwelt Baden-Württemberg
Today's software products usually consist of individual subsystems which are developed by individual teams of developers and combined to a software system. Interfaces between the teams often have to be communicated. A good communication and documentation contributes significantly to the success and sustainability of software projects. Especially in larger projects it is important to share the relevant information and make it easily accessible for all developers involved. A developer portal represents a central point about all services and their interfaces.

(1) The goal of the project is to create a developer portal, which is specifically designed to meet the requirements of today's agile development of cloud native software. The developer portal should be built on the existing development process of C&M and transfer the information into the developer portal via the CI/CD pipeline automation. In addition to the provision of the API, further information about the service such as metadata, dependencies towards other services and the status of the running services should be displayed via the developer portal.

(2) During the internship, individual tickets are defined, which are discussed and evaluated together in a story estimation. Subsequently, the previously discussed tickets can be processed by the students. To achieve the project goal, the following work packages should be implemented.

(2.1) Existing solution described in publications or documented solutions of existing products (e.g. API management tools) are to be analyzed to get ideas for the requirements and capabilities in this project.

(2.2) Before the start of the design phase, the students should review all relevant data provided by the services and artifacts of the C&M development process. Missing data in the project organization in GitLab should be added and/or the process in the project organization in GitLab should be adjusted to provide the appropriate data in the future.

(2.3) After all relevant service data has been determined, the C&M development process for the developer portal should be carried out. This includes the Behavior Driven Development (BDD) as well as the Domain-driven Design (DDD) and the subsequent development of an API.

(2.4) Using the artifacts created previously, the developer portal can be implemented in any desired programming language. Like the other services, the developer portal should run within the Kubernetes cluster. Therefore a Helm chart should be created and deployed as a cluster service.

(2.5) In order to provide the service data, the data must be sent to the developer portal each time the CI/CD Pipeline is executed. This requires an adjustment of the globally used pipeline templates from C&M. The extension should transfer the relevant service data determined in (2.1) and (2.2) to the API interface of the created developer portal.
Cloud-based solutions often add additional complexity to the development of software systems. Due to the additional layers of abstraction like Docker and Kubernetes, a developer requires additional skills and knowledge of operational concepts. To reduce the complexity of the additional layers, we introduced the project templates, which provide the developer an initial starting point for the development of cloud-based applications. Each of the project templates provides a separate Helm chart. Therefore, the maintenance has to be done for all project templates separate. A shared Helm chart template should overcome this issue since it provides a capability for the integration of additional features.

1. Helm charts are by default agnostic. For this reason, a central managed and shared Helm chart can be shared between the framework-specific project templates. The sharing of the Helm chart template reduces the effort for maintenance and integration of new features. To validate the benefits of a central Helm chart, the chart should be extended to allow the integration of external services of C&M.

2. Preparation is a major part of this project team, to ensure the implementation suits the requirements of the developers.
   2.1 A basic understanding of Helm and its capabilities is required for a good design of the Helm chart template. Therefore, a basic tutorial will be provided, where students learn the basics of Helm.
   2.2 JuniorStudents learn the basics of the shared pipeline and project template approach.
   2.3 In previous work, problems in the implementation of the template approach were analysed. In order to avoid further issues, the problems should be taken into account in the creation of the central Helm chart.
   2.4 From the insights gained, the requirements for the chart and the goals for implementation should be defined.
   2.5 The requirements should be implemented accordingly to the previous work and a central Helm chart should be stored within Git and the registry. To ensure the the correctness of the chart, test should be provided.
   2.6 The created chart template should be integrated and tested in the existing project templates.
Requirements for Participating in the Practical / Seminar Course

1. Project team meeting each week
   (1) The dates are individually fixed by the project team leaders and the Junior Students

2. Continuous Writing (CW) and delivery of versions of the practical / seminar thesis (English/German, LaTeX)
   (1) Continuously Written practical/seminar thesis versions are Continuously Reviewed (CW/CR) by the co-supervisors

3. Students who want to pass both a practical course and a seminar course should have completed their practical course (= 150 working hours) at least 5 weeks before the end of the lecture period

4. Presentation of the results in the WASA lecture
   (1) See dates for each project team on the page <<Lecture Plan>>

5. Delivery of the final practical /seminar thesis on the last day of the lecture period

This page summarizes all relevant dates that are relevant for each participant of the practical /seminar course offered in combination with the WASA lecture.

1. It is absolutely necessary that a student has a free slot in his/her personal time table in order to be able to take part in these weekly meetings. The meetings start in the next week (i.e. the second week of the lecture period).

2. This preliminary version is reviewed by the co-supervising PhDResearcher/SeniorStudent.

3. The precise date is communicated in the project teams. The date is also fixed in the team calendar of the C&M Teamserver.

4. The content produced for the presentation should conform to the WASA course material.

5. The delivery date is the last day of the lecture period.
Next Steps

(1) Personal decision if you want to participate in the WASA lecture and an accompanying practical/seminar course

(2) If YES

(1) Latest until Thursday, 05.11.2020, 10 am: Send an email with your depseudonymized KIT mail address to sebastian.abeck@kit.edu and michael.schneider@kit.edu with the following information:

(1) Prename, surname, matriculation number, interest in practical and/or (pro)seminar course

(2) Personal motivation and experiences in this area (e.g. IT project experiences)

(3) 2 to 4 project teams of interest (highest priority first)

(3) The answer of your email will contain all relevant information to start the lecture and the practical/seminar course

(1) Access to team server, date of kickoff meeting, first work packages

(1) It is absolutely important that a student who participates in the WASA lecture and practical/seminar course has the necessary resources to cope with the workload (lecture: 120 hours, practical course: 150 hours).

(2.1) Check on the page of the KIT Steinbuch Computing Centre if your email is already depseudonymized.

(2.3) The motivation and the experiences should be summarized in at least one or two paragraphs.

(2.4) This means that the first project team in the list is your favorite team.

(3) This email will be sent by the co-supervisor of the members of the project team which usually is a SeniorStudent.

(3.1) The current lecture material is stored on the C&M Teamserver in the following folder: https://team.kit.edu/sites/cm-tm/Mitglieder/2-0.Aktuelles_Semester
Each project team makes a specific contribution to the show case.

(PT ZieschinskiKlechorov) This PT is concerned with the PredictiveCarMaintenance (PCM) application for which a complete test concept including all types of test (unit, integration, end-to-end) should be developed.

(PT MetinMüller) The development of the ClinicsAssetManagement (CAM) application with the focus on medical devices is part of PT MetinMüller.

(PT Schneider) The goal within PT Schneider is to provide Internet of Things (IoT) aspects for the ClinicsAssetManagement (CAM) application.

(PT HippchenSidler) The goal of PT HippchenSidler is the development of design artifacts for an application in the environmental context.

(PT Rudolf) The goal of PT Rudolf is to apply machine learning methods in the context of IAM and security.

(PT SängerHanselmann) The goal of PT SängerHanselmann is to develop a developer portal which supports the efficient exchange of developers and the monitoring of the existing services.

(PT Throner) Within the project a Helm chart template should be created, which fulfills the common requirements for software development.
The C&M Teamserver is described in detail in the document C&M-TEAMARBEIT (in German) which is available
(i) on the C&M web site: https://cm.tm.kit.edu/
(ii) on the C&M Teamserver: https://team.kit.edu/sites/cm-tm/Mitglieder/1-1.Teamarbeit
Activation of the Name-Related E-Mail Address (Depseudonymization)

(1) Can be carried out via Shibboleth (https://my.scc.kit.edu/shib/pseudonymisierung.php)

This function can be found in the Studierendeportal by clicking on "Meine Benutzerdaten" > "De-/Pseudonymisierung" and accepting "Ich stimme der Sichtbarkeit meiner namensbezogenen Daten zu". Die name-related E-Mail-Adresse <prename><surname>@student.kit.edu" exists additionally to the "uxxx@student.kit.edu" email address.