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 winter semester 2020/21, the lecture and practical/seminar course are organized as an online event. This new format is called WASAOnline.

The WASAOnline kickoff lecture will take place on Wednesday, 4th November 2020 on 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.abec@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, Internet of Things) In this area connected car application and IoT 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).

(Identity and Access Management, SecDevOps) SecDevOps concerns the continuous integration (CI) and continuous deployment (CD) of (hopefully not monolithic, but microservice-based) software systems deployment into a container-virtualized (Docker/Kubernetes-based) cloud infrastructure. The Identity and Access Management (IAM) is a highly relevant crosscutting concern appearing in almost every web application. One of the projects is concerned with the provisioning of IAM as a flexible cloud service.

<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>IOSB</td>
<td>Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>SecDevOps</td>
<td>Security Development 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
The activity diagram gives an overview of the whole engineering process which in fact is divided into two sub-processes: According to our understanding of DDD, a domain model is the foundation of all software applications belonging to this domain. We regard the domain model as the knowledge of a domain expert he or she makes available to the software developer by the model. This knowledge is the application-agnostic part of the application meaning that each application of the domain implements this knowledge in its domain logic layer. We understand this domain knowledge as the "physics" that each application must adhere to. Evans calls this part of functionality the heart of the software system [Ev03].

(Starting Points) (Strategic Modeling of the Initial Context Map) The starting point of the domain modeling process is set earlier as the starting point of the application development process. The reason is that it makes sense to have an initial context map of the domain before the development of the first application starts. In the following we describe the main characteristics of the application development process.

(Context Map) The main artifact of the domain modeling process is the context map in which all micro-service-based applications belonging to the domain must fit into.

(Strategic Modeling of the Bounded Contexts as Part of the Domain's Context Map) In the application development process, the integration of the application under development into the context map of the domain is done in the second step after the first step in which the BDD-based requirements analysis is carried out.

(Tactical Modeling of a Bounded Context) A bounded context is part of a context map. It represents a candidate for a microservice which can be developed by an independent team [Ne15]. DDD considers the modeling of the content (i.e. the functionality) of a bounded context as tactical modeling.

(Relation View) The most relevant artifact of the tactical modeling is the relation view which is built in the third step. By the relation view central DDD concepts (such as entity and value objects) including their relationships and operations are represented.

(Implementation of each Bounded Context as a Microservice Accessible via its Web API) The functionality modeled by the relation view is implemented in the domain logic layer of the microservice architecture. A systematic approach to implement the bounded context as a microservice (fourth step) is a core characteristic of the proposed development process. One of the key demands of DDD is to keep the model and its implementation in sync in order to assure that the model and the implementation is the same.

The two remaining steps concern development activities around the core implementation of the microservice, namely the implementation and test and the deployment of the application.

The main dimension according which the development tools can be ordered are the development phases (from analysis to deployment). The tools supporting the analysis and design and the implementation and testing can be grouped according to the software architecture (which in our case is a microservice architecture).

(Project Management and Version Control) These tools support the overall organization of the software project and the communication between the project members. At C&M two different tools sets are used: the Atlassian toolset extended by Microsoft tools, esp. SharePoint on which the C&M Teamserver is based.

(Analysis and Design) Analysis requirements at C&M is done by taking the approach of behavior-driven development (BDD) based on the tool Cucumber. For the design the two most relevant tools are Enterprise Architect for the domain model and Swagger for the API specification. In addition to these tools the Microsoft Office tools (Word, PowerPoint) and Atlassian Confluence (in the case of iCC/xdi projects) are applied for documentation purposes.

(Implementation and Testing) In this phase the frontend and backend of the web applications are constructed. The integrated development environment (IDE) used for frontend development is JetBrains' WebStorm and frontend frameworks are Angular and Bootstrap. As IDE for backend development Eclipse is used and Apache Spring (esp. Spring Boot for the microservice implementation) is used as backend framework.

(Build and Deployment) The build and deployment of the microservices is carried out via a build pipeline by which the concept of continuous integration and continuous deployment is provided. The result of the build pipeline is a Docker image (= application container image) since Docker is used as the container environment at C&M. Kubernetes is used to manage the Docker containers in order to reach a high scalability and robustness of the service landscape.

BDD Behavior-Driven Design
IDE Integrated Development Environment
(PredictiveCarMaintenance, Connected Car) The whole case study centers around the PredictiveCarMaintenance (PCM) which is a microservice-based application from the connected car domain.

(User, PredictiveCarMaintenance, PCM) The connected car application implements the predictive maintenance of a vehicle. The functionality of the PredictiveCarMaintenance (PCM) is specified by Gherkin features.

(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.
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.
# WASA Modules and Courses

<table>
<thead>
<tr>
<th>Module Type</th>
<th>Description</th>
<th>Credits/Hours</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASA Lecture</td>
<td>WASA1 for &quot;Informatik / Wirtschaftsinformatik Bachelor&quot; students in winter semester</td>
<td>4 credits</td>
<td>The lecture can only be visited in combination with a WASA practical and/or proseminar/seminar course taken in the same semester.</td>
</tr>
<tr>
<td></td>
<td>WASA2 for &quot;Informatik / Wirtschaftsinformatik / Informationswirtschaft Master&quot; students in summer semester</td>
<td>4 credits</td>
<td></td>
</tr>
<tr>
<td>Oral examination</td>
<td>20 minutes in German</td>
<td></td>
<td>In the last week of the lecture term.</td>
</tr>
<tr>
<td></td>
<td>in the lecture term</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## 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) proseminar 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 PhDResearchers. 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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>The practical/seminar work is carried out in a project team</td>
</tr>
<tr>
<td>(1.1)</td>
<td>A student doing his/her practical work at C&amp;M takes the role of a JuniorStudent co-supervised by a SeniorStudent/PhDResearcher</td>
</tr>
<tr>
<td>(2)</td>
<td>The team meets each week for about one hour</td>
</tr>
<tr>
<td>(3)</td>
<td>JuniorStudents work together with the co-coaching SeniorStudent/PhDResearcher on specific topics that are relevant for his/her Bachelor/Master/PhD Thesis</td>
</tr>
<tr>
<td>(3.1)</td>
<td>JuniorStudent continuously provide contributions that are discussed in the team (Continuous Writing CW)</td>
</tr>
<tr>
<td>(3.2)</td>
<td>Reviews by the SeniorStudent/PhDResearcher provide helpful feedback (Continuous Review CR)</td>
</tr>
<tr>
<td>(3.3)</td>
<td>The work packages defined during the practical/seminar course depend on the results and knowledge gained from the work done so far</td>
</tr>
<tr>
<td>(4)</td>
<td>Own ideas how to deal with a topic are welcome</td>
</tr>
</tbody>
</table>

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.
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 show case.

(PT SchneiderMetinMüller)

(PT Sidler)

(PT Hippchen)

(PT ThronerSänger)
The goal within this project team is to provide the position data of devices which are used in medical clinics.

1.1 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 the Internet of Things (IoT). Therefore, the sensor information is standardized using the SensorThings API from the Open Geospatial Consortium (OGC).

1.2 A widely used data standard in the healthcare sector is Fast Healthcare Interoperability Resources (FHIR), which describes a standardized data format for the digital exchange of data in the healthcare context. In the project work, the standard is used to describe the medical data. Furthermore, the devices metadata is stored in the standard.

In the following, the work packages of the IoT project team are described.

2.1 Both of the standards are used to store different types of data. Data which is concerned to IoT is stored by the SensorThings API and medical data and device metadata is stored in the FHIR standard. This separation of the business and IoT concerns is one of the first tasks of the project team.

2.2 The separation of the IoT and business concerns need to be expressed in the domain models of the healthcare and IoT domain.

2.3 The IoT information such as sensor information needs to be mapped to the SensorThings API standard. Fraunhofer's FROST-Server is used as SensorThing API implementation. A FROST-Server instance already exists to fulfill this work. The medical and metadata needs to be mapped to the FHIR standard (similar to the mapping of the IoT data).

2.4(2.5) One goal is to store the mapped medical and metadata in the FHIR standard. Therefore the API needs to be analyzed and understood to send the data to the FHIR server. For this task, a FHIR server needs to be set up and configured. The SensorThings API access was already analyzed in previous work.

CAM  ClinicsAssetManagement
FROST-Server  Fraunhofer Open Source SensorThings API Server
IoT  Internet of Things
OGC  Open Geospatial Consortium
FHIR  Fast Healthcare Interoperability Resources
In the healthcare domain, a variety of different legacy software systems already exist and are used in clinics. The knowledge of the software systems shall be used in the context of the application Clinics Asset Management (CAM). Therefore, a consideration of these software systems is one goal of this project team.

Another goal of the project team is to design and implement the CAM microservices based on the knowledge gathered from the legacy software systems. For the implementation, a collaboration with the project teams PT Sidler and PT Schneider is necessary.

First, the existing legacy software systems must be understood. Relevant here are the behavior-specific aspects of the software systems that meet the functional requirements. In the end, the domain objects can be identified.

The structural aspects of the legacy software systems, which also result from the behavioral aspects, must be captured by tactical modeling. This is because the CAM microservices to be developed should be based on the domain knowledge. In particular, the structure of the domain objects plays a major role here.

The task requires a high level of collaboration with the two project teams PT Sidler and PT Schneider. For the backend for frontend, the interfaces have to be coordinated. Furthermore, the interfaces of PT Schneider must be considered for the consumption of the IoT data. If necessary, changes due to new requirements must be discussed with the project teams. Gradually, the designed microservices are implemented. The chosen programming language is Golang.

API Application Programming Interface
CAM Clinics Asset Management
IoT Internet of Things
The goal of this project team in the context of the ICD2020 is the design of a BFF microservice that supports the overall PCM architecture by merging request calls from the PCM frontend to the backend microservices. Therefore, the complexity of the frontend is simplified by isolating the data formatting in the BFF and by reducing the number of requests the frontend has to perform.

1. A suitable BFF API has to be defined. It should closely relate to the functionality of the frontend.
2. The BFF receives requests from the frontend and processes them by sending requests on the backend microservices.
3. The BFF retrieves data from the backend microservices and delivers this data in a desired format to the frontend.

(2) In the following, work packages for the project team are defined.
1.1 The first task is to understand all features and functionality of the frontend. Based on this analysis, necessary endpoints are identified.
1.2 Based on the identified endpoints in (2.1), the API has to be specified including URLs, query parameters and data formats. The API must be a REST API and the specification should be designed by using the tool OpenAPI.
1.3 The BFF is implemented by the students based on the previous tasks.
2.4 A further task is the implementation of a concept to serve the different roles that exist within the PCM application. Depending on these roles, the frontend offers different views and therefore, different data from the backend has to be fetched and returned to the frontend.
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 service registry represents a central point about all services and their interfaces.

(1) The goal of the project is to create a service registry, which is specifically designed to meet the requirements of today's agile development of cloud native software. The service registry should build on the existing development process of C&M and transfer the information into the service registry 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 portal of the service registry.

(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 documentations of existing products (e.g. API management tools) are to be analyzed to get ideas for the service registry designed and implemented in this project.

(2.2) Before starting the design, 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 development of the service registry 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 Service Registry can be developed in any desired programming language. Here the service should be executable, like all other services within the Kubernetes cluster. For the delivery to the Kubernetes cluster Helm should be used.

(2.5) In order to provide the service data, the data must be sent to the service registry 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 service registry.
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.
(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
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". The name-related E-Mail-Adresse <prename><surname>@student.kit.edu" exists additionally to the "uxxx@student.kit.edu" email address.
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