This WASA course unit [CM-W-WAS] 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).

WASA1 (Bachelor): 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-Cl, 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 different domains (Healthcare, ConnectedCar) and includes concepts from the domain of Internet of Things.

WASA2 (Master): A compact summary of the concepts covered by WASA1 is provided. In WASA2, two advanced topics are focused: (i) API engineering and management and (ii) identity and access management. Both topics are highly relevant for the digitization strategy of companies. In the lecture, two leading products (MuleSoft, Okta) are introduced to illustrate how the topics are solved in IT practice.

Since the concepts presented in the lecture must be practically applied to really understand the concepts, the WASA lecture is only offered in a combination with the WASA practical course.

_The WASA kickoff lecture will take place online on Wednesday, 26th October 2022 at 9:45 am._

Each student who wants to take part in WASA and in the kickoff lecture should send an email to cm.research@lists.kit.edu to receive the web link to the web session which is made available for their personal use. Please do ONLY use your DEPSEUDONYMISED 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 lecture, the content is presented and discussed in German. The oral examination is conducted exclusively in German. All students write their practical/seminar thesis in English. Thesis templates are made available in LaTeX. Overleaf is used for the collaborative writing of the practical thesis.

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

[CM-W-WAS] Cooperation & Management: WASA INTRODUCTION. \sccfs.scc.kit.edu/OE/TM/VR\Mitglieder\2-2_WASA_Lecture
The research work, carried out by C&M, can be divided into two main areas:

(Connected Car, Healthcare, Environmental) In these business domains, applications based on the concept of domain modeling and microservice architectures are developed. Relevant concepts applied in the microservice engineering approach include Behavior-Driven Development (BDD) and Domain-Driven Design (DDD). In addition to development (Dev), the operational aspects (Ops) are intensively taken into account. DevOps concerns the continuous integration (CI) and continuous deployment (CD) of microservice-based software systems into a container-virtualized (Docker/Kubernetes-based) cloud infrastructure.

(Microservice API Integration and Management, Identity and Access Management) API Engineering and Management can be seen as the bridge between the two C&M’s research areas since its task is to integrate and manage (i.e., configure, monitor, control) the developed APIs. Identity and Access Management (IAM) is a highly relevant crosscutting concern appearing in every web application. An advanced approach is IAM as a Service (IAMaaS) by which the IAM functionality is 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>DevOps</td>
<td>Development and Operations</td>
</tr>
</tbody>
</table>
A microservice architecture is located on the application plane as it is introduced in ++Network, System, and Application Plane++. While the software architecture is described by the logical layers specified by a specific DDD pattern LAYERED ARCHITECTURE, the system architecture introduces several subsystems (domain microservices, application microservices, API gateway). Two types of application programming interfaces (domain microservice API, application microservice API) are separating the logical layers on the software architecture side and the microservices on the system architecture side.

(Presentation Layer) This layer renders the UI elements in the browser. Technologies that support the implementation are Angular and Bootstrap. The presentation includes a logic which controls the interaction with the application microservice API. An optional API gateway is often used to provide cross-cutting concerns, such as load balancing or security aspects.

(Application Logic Layer, Application Microservice) This layer realizes the orchestration of domain microservices in order to provide the application logic to fulfill the requirements made to the application. A technology that supports the implementation of this functionality is Spring.

(Domain Logic Layer, Domain Microservice) This layer implement the domain microservices which mainly are CRUD operations on the domain objects.

(1) In contrast to a traditional three-layer application architecture, the business logic layer in a microservice architecture is split into two layers, the domain logic layer and the application logic layer. The reason for that is to promote the reuse of business logic functionality by distinguishing between application-agnostic (= domain logic) and application-specific (= application logic) functionality.

API Application Programming Interface
CRUD Create, Read, Update, Delete
The diagram describing a task process is explained with the example of the user/system interaction "Register a Medical Device" as it was introduced in the chapter ANALYSIS.

(GUI Register a Medical Device) The Graphical User Interface (GUI) contains all interface elements that are needed to carry out the user/system interaction. In addition, presentation logic provides the interworking of these elements and the communication with the application microservice via its API.

(1/0) The application microservice operation is called by the gRPC function RegisterAMedicalDevice which was introduced on the page ++gRPC API of the Application Microservice MedicalDeviceManager++.

(1. U ...) This first step of the user/system interaction has no effect on the task process. Therefore, a number /1/ is missing in the task process.

(2. S ...) This step results in the task Control Access /2/.
/2.1/ /2.2/ REST operation provided by the domain microservice Staff
/2a/ Error message (access not allowed) of the gRPC function

(4. S: ...) The task Check Input /4/ makes checks which are not included in the checks carried out by the domain microservice.
/4a/ Error message (error in medical device information) of the gRPC function

(5. S: ...) A successful call /5req/ /5rsp/ of the task Register Device to the domain microservice MedicalDevice leads to a successful output message /5/ of the application microservice operation RegisterAMedicalDevice. In the case of an error an error output message /5a/ is the result.
The software architecture of CCSApp is based on MuleSoft's integration architecture which is extended by a domain logic layer (CM-G). While the design of the application logic layer is derived from the analysis of the applications (esp. the use cases), the design of the infrastructure layer is prescribed by the external systems which must be integrated.

1) For this kind of User Interface (UI), the Experience API E-CCSAppWeb provides the access to the application logic realized by the Process APIs. If another type of UI is needed (e.g., a mobile app), the presentation layer will be extended by a second Experience API.

2) In the MuleEng approach, each capability leads to one Process API and each use case of this capability leads to one endpoint of this Process API. The systematic derivation process will be described on one of the following pages.

3) The external systems Ext-DaimlerCar and Ext-BMWCar provide static and dynamic information in different ways which lead to two different System APIs S-DaimlerCar and S-BMWCar. The Domain API maps the heterogeneous data from the two System APIs to the description of a car as it is defined by the domain model ConnectedCar.

4) The CCSApp does not include the rental functionality, but extends the rental service by more sophisticated functionality for which the information from the car APIs is needed.

UI User Interface

/blob/master/pages/ccsapp_softwareArchitecture.md
ClinicsAssetManagement CAM, ConnectedCarServicesApplication CCSApp These are the microservice-based applications which are developed by C&M in cooperation with industrial cooperation partners and partner institutes.

IAM Service) The functionality needed by each application include Identity and Access Management (IAM) requirements, such as authentication of the user of the application. A relevant IAM solution to implement these requirements is Okta.

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

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

IoT, IoT Service, SensorThings, Web of Things) IoT standards, such as SensorThings API from OGC and Web of Things from W3C, are taken into account in the development of the IoT-based application.

API Security) The APIs 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) The goal is to develop a build pipeline based on GitLab-CI by which the application is continuously integrated (CI) and deployed (CD) on a Kubernetes cluster. The microservice-based applications provide metrics which are monitored tools such as Prometheus.
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) 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 (German: Leistungspunkt).

(2) The practical course and the proseminar/seminar run in parallel with the lecture course. The practical course counts 5 credit points meaning a workload of 150 hours and the proseminar and seminar count 3 credit points meaning a workload of 90 hours.

Remarks:
(i) If the number of WASA applications is high, those students are preferred who want to pass the practical course.
(ii) 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 PhD 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.

<table>
<thead>
<tr>
<th>WASA Modules and Courses</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) WASA Lecture</td>
<td></td>
</tr>
<tr>
<td>(1) WASA1 for &quot;Informatik / Wirtschaftsinformatik Bachelor&quot; students in winter semester</td>
<td></td>
</tr>
<tr>
<td>(2) WASA2 for &quot;Informatik / Wirtschaftsinformatik / Informationswirtschaft Master&quot; students in summer semester</td>
<td></td>
</tr>
<tr>
<td>(3) Oral examination of the WASA lecture</td>
<td></td>
</tr>
<tr>
<td>(1) 20 minutes in German</td>
<td></td>
</tr>
<tr>
<td>(2) Last week of the lecture term</td>
<td></td>
</tr>
<tr>
<td>Wased1 Lecture (4 Credits)</td>
<td></td>
</tr>
<tr>
<td>WASA1 Practical Course (5 Credits)</td>
<td></td>
</tr>
<tr>
<td>or Proseminar (3 Credits)</td>
<td></td>
</tr>
<tr>
<td>Module Key Qualifications (&quot;Schlüsselqualifikation&quot;)</td>
<td></td>
</tr>
<tr>
<td>Teamwork in the Area of WASA (2 SQ Credits)</td>
<td></td>
</tr>
<tr>
<td>WASA2 Lecture (4 credits)</td>
<td></td>
</tr>
<tr>
<td>WASA2 Practical Course (5 Credits)</td>
<td></td>
</tr>
<tr>
<td>or Seminar (3 Credits)</td>
<td></td>
</tr>
</tbody>
</table>

SQ Schlüsselqualifikation (Key Qualification)
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 flexibility and liberty concerning the focal points of the practical/seminar work.

(3) Therefore, different templates are made available for a practical course thesis and a seminar thesis.
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 of 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 practical thesis and the 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) PredictiveCarMaintenance, ClinicsAssetManagement, Microservice Engineering, IoT, SensorThingsAPI, Data Protection

(PT Sänger) MicroserviceDeveloperPortal, API Management, IAM

(PT Throner) DevOps, DevOps Templates, Pipelines, Kubernetes Cluster
The goal of subteam1 of PT Schneider is to provide the current position of a medical device to the ClinicsAssetManagement (CAM) application. For this purpose, the position of the medical device must be known. If a device is moved to another location, the current position must be adjusted. Sensors of a medical device provide the geolocation. This geolocation format needs to be mapped to the required data format of CAM. For this purpose, the microservice SensingDevice was developed in the previous semester. In the next step, the application microservice MedicalDeviceLocator requires to be developed. Especially a frontend needs to be developed.

The IoT functionality should cover analytics and logging of the domain IoT. For example, it is interesting how many information is gathered from the sensor in the last hour. The goal of the project team is to develop one of those services.

Subteam2 (Zingg) is concerned with developing the capability "Management of Vehicles". One task contains the development of the capability management of vehicles by following the microservice engineering process. For the implementation, the implementation and test concept (with focus on the tests) is applied. Therefore, application constraints are defined which are used to derive tests. Problems and insights should be stated and suggestions for improvements are part of this task.

Especially integration, consumer-driven contract tests, and end-to-end tests should be implemented according to the concepts if the implementation and test concept. It is important, that the tests run in a CI/CD pipeline. Furthermore, cross-triggering of the pipelines is required which is implemented by the project team.

The task of subteam3 (Ablaßmeier) is to ensure that the implementation follows the microservice engineering approach, and especially the software quality during the implementation phase.

As a result, the structure of the code (source folder) and the code quality need to be automatically ensured. Therefore, SonarQube should be used as a tool to ensure the quality. Next, the tool should run in a GitLab pipeline to ensure that the repository still follows the guidelines after each commit.

Next to the structure, code quality should be measured as well. This part should also be realized with SonarQube. For example, the quality should ensure that method names are meaningful and if common check style (e.g., eslint) are properly used during the implementation (and the code). The results are to be recorded and discussed in the project team. What software quality includes is covered in a (pro-)seminar.
(1) Identity and Access Management (IAM) is not yet in practical use at C&M. The goal of the subteam1 (Ricker) is to add authentication and authorization to the MicroserviceDeveloperPortal (MDP). At C&M, Keycloak is used as an IAM system.

(1.1) The authentication and authorization methods must be added to the frontend and backend services of MDP. This includes a login flow using OpenID Connect (OIDC).

(1.2) The goal is to develop a cross-cutting domain IAM which can also be used by other applications. The gained insights of the subteam Ricker in adding IAM functionalities to an application are then used for a first draft of a cross-cutting domain IAM. One seminar goal can be to research IAM functionalities and support the modelling of a cross-cutting domain IAM.

(2) Subteam2 (Skrodzki) is concerned with the further development of the API management capabilities of the MDP.

(2.1) The MDP includes a first implementation of the capability "Management of APIs" which is untested and unfinished. Currently, the MDP can only register and unregister an API.

(2.2) The configuration of the API gateway Tyk through the MDP must be implemented.

(2.3) A frontend for the capability "Management of APIs" to manage API versions and an API gateway configuration must be added to the MDP.

(3) Subteam3 (Hoger) deals with the extension of the CI/CD pipeline for the MDP. The pipeline extension is called RegisterStep and is used to register new microservices and APIs to the MDP.

(3.1) The current version of the RegisterStep requires an access token for a GitLab repository to register a microservice. This must be changed in a future version of the RegisterStep. Furthermore, the placement of the RegisterStep in the MDP and the ServiceEnvironment has to be further investigated.

(3.2) The placement of the RegisterStep comes along with the revision of the modelling of the domain ServiceEnvironment. This includes rethinking the current subdomains and the placement of API management functionalities.

API Application Programming Interface
IAM Identity and Access Management
MDP MicroserviceDeveloperPortal
OIDC OpenID Connect
Cloud services are currently one of the biggest accelerators for digital transformation. Due to the classic separation of development and operations, the provisioning of these is still often carried out separately from development. This can lead to problems in the coordination and configuration of the services. Accordingly, the goal of subteam 1 (Filippov) of project team Throner will be the integration of cloud services into the software development process.

(1.1) For the provisioning of cloud services, the infrastructure-as-code approach is mostly used. Within the scope of the internship, common tools for this purpose will be investigated and compared.

(1.2) In order to integrate cloud services into the development process, a process needs to be defined which allows developers an easy configuration and integration into their services. As a base the template-based DevOps approach from C&M should be used.

(1.3) For the domain PredictiveCarMaintanance a managed database from a cloud provider should be used.

(2) Internet facing applications provide a large attack surface for hackers. To prevent potential attacks, it is important to regularly update the application and close potential vulnerabilities. The subteam 2 (Petrovic) of the project team Throner deals with the automatic detection and elimination of these vulnerabilities.

(2.1) In order to detect security vulnerabilities in the system, it is necessary to read the image of the containers and the dependencies installed within the containers and compare them with a Common Vulnerabilities and Exposure (CVE) database. Each CVE describes a vulnerability and its impact on security. If there is a patch for the vulnerability, it will also be listed in the CVE.

(2.2) To make developers aware of the vulnerabilities, all vulnerabilities a microservice has should be displayed in the MicroserviceDeveloperPortal.

(2.3) An automated process shall be implemented which automatically fixes the vulnerabilities in the running systems. To do this, all possible patches must be extracted from the previously persisted security vulnerabilities and transferred to the source code of the services.

AWS Amazon Web Services
CVE Common Vulnerabilities and Exposure
FHIR Fast Healthcare Interoperability Resources
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) The content produced for the presentation should conform to the WASA course material.

(3) This day is the Friday of the last lecture week. The whole Friday (i.e. 12 pm) is available to finish the documentation.
**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, 27.10.2022, 10 am:** Send an email with your depseudonymized KIT mail address to cm.research@lists.kit.edu with the following information:
      1. Prenom, surname, matriculation number, interest in practical course (preferred) or (pro)seminar course or both
      2. Personal motivation and experiences in this area (e.g., IT project experiences)
      3. 2 to 4 project teams of interest (highest priority first)
   2. The answer to your email will contain all relevant information (esp. access to team server, date of kickoff meeting, first work packages) to start the lecture and the practical/seminar course

(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.

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

(2.1.3) 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.

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
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
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.