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

(Connected Car, Healthcare) In these two business domains, applications based on the concept of domain modeling and microservice architectures are developed. Relevant concepts applied in the microservice engineering approach include Domain-Driven Design (DDD) and API (Application Programming Interface) lifecycle management. 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. An Integrated Platform as a Service (IPaaS) supports the whole API lifecycle management. 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.

**Abbreviations**

- API: Application Programming Interface
- CI/CD: Continuous Integration / Continuous Deployment
- DDD: Domain-Driven Design
- IAM: Identity and Access Management
- IPaaS: Integrated Platform as a Service
- DevOps: Development and Operations
The software architecture of the application ConnectedCarServiceApplication (CCSApp, in concrete CCSAppV0.1) [CM-G-Sof] is based on the Mule microservice Architecture, MuleArch. 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.

CCSApp ConnectedCarServicesApplication
UI User Interface

In the research group C&M the application CCSApp serves as one of the demonstrator for the research work carried out in the areas of (MuleSoft-based) microservice engineering, access policies, and DevOps. For each product increment a precise specification of requirements is defined the resulting CCSApp should meet.

(1) The work on the application CCSApp started in May 2022. At that time, a focus of the C&M research work was on a microservice engineering approach based on MuleSoft (i.e., MuleEng).

(1.1) (1.2) The base use cases were partly implemented as a preparation of the IAM Coding Day (ICD) whereas parts of the challenge use cases were implemented during the ICD event.

(1.3) The result of this work is a broad set of analysis, design, implementation, and deploy artifacts – but still more or less incomplete patchwork.

(2) The artifacts from CCSAppV0.1 served as a valuable input for CCSAppV1 which can be understood as the minimum viable product of the application CCSApp.

(2.1) The main driving force were the SeniorStudents who started their Master Thesis in Summer Semester 2022.

(2.2) The idea of CCSAppV1.0 is to reduce the functionality to a minimum.

(2.3) The guidelines and best practices especially concern the design and the implementation of the APIs and the successful deployment of the implemented APIs.

(2.4) Therefore, CCSAppV1.0 is the first product increment of the application CCSApp.

(3) (3.1) The development of the second product increment, CCSAppV2 is the central goal of the three project teams working in Winter Semester 2022/23 at C&M.

(3.2) All project teams will work on the analysis, design, implementation, and deployment of the base use cases.

(3.3) The development of CCSAppV2.1 will be carried out during the third phase of the practical course (realization phase, Umsetzungsphase). During the second phase (concept phase, Konzeptphase), this work is planned by the project team leaders (i.e., PhD Researchers) together with the Senior Students.

- P1.MicroserviceEngineering: (i) Extension of the functionality of CCSAppV2.0 by one further challenge use case, (ii) Systematic testing
- P2.AccessPolicies: (i) Extension of the CCSAppV2 architecture by an externalized authorization based on policies, (ii) Derivation of access policies from analysis artifacts and implementation of these policies
- P3.DevOps: (i) Automated deployment based on the CloudHub API, (ii) Use of the Runtime Fabric
Two microservice-based applications, ClinicsAssetManagement (CAM) and ConnectedCarServicesApplication (CCSApp) are developed by C&M in cooperation with industrial cooperation partners and partner institutes.

Two approaches to develop microservice-based application are followed by C&M: (i) The C&M Engineering (CMEng) approach which is strongly based on the DDD concepts and (ii) the Mule Engineering (MuleEng) approach which focus is on the support of the API lifecycle management including the integration of external systems via microservice APIs.

The functionality needed by each application include Identity and Access Management (IAM) requirements, such as authentication of the user of the application.

Internet of Things (IoT) standards, such as the SensorThings API, are taken into account in the development of the IoT-based application.

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.

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 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 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)
The figure illustrates the cooperation of the different roles of the C&M researchers. As a result of this intensive cooperation, the resulting documents provide valuable to documents which are on a higher hierarchical level.

(WASA Lecture) The WASA lecture summarizes the current research results of C&M. This content builds the fundament of C&M’s research work and must be known by all C&M researchers.

(JuniorStudent, SeniorStudent) Student who take part in the WASA lecture and who write a practical (or seminar) thesis take the role of a JuniorStudent. When they decide to write a bachelor/master thesis at C&M they change the role and become a SeniorStudent.

(supervises, co-supervise) C&M follows a clear and wellthought-out supervision concept which includes the definition of the weekly meetings and the process how and when to review contributions of the different roles.
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/Head
   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 carried out so far
   4. Own ideas how to deal with a topic are welcome
3. The focus of the practical course is on practical IT engineering whereas the proseminar/seminar concentrates on the analysis of scientific publications

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.
Working in a Project Team

1. Each project team meeting is prepared by an agenda to which all team members contribute.
2. Markdown document in GitLab
3. Weekly project team meetings starting next week
4. Project team members communicate in German
5. Junior Students write their practical/seminar thesis in English and use the LaTeX tool Overleaf to format their text
6. 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 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 the agenda of project team meetings.

\sccfs.scc.kit.edu\OE\TMVR\Mitglieder\1-1.Teamarbeit
Each project team makes a specific contribution to the overall microservice-based system environment.
(1) Subteam 1 tackles persistency aspects. The goal of this project team is to create, provide, and implement a persistency concept for CCSApp by enhancing the engineering approach. A question to answer is which of the microservices require a database. One important property of such a persistency concept is the horizontal scalability of the implemented microservices. (1.1) Both engineering approaches (CMEng and MuleEng) use a micro architecture for implementation. This micro architecture should be extended to provide a persistency concept, as well as the formatting of the data.
(1.2) Several possibilities for implementing databases exist. Examples are the following: private tables per service, a schema per service, and an own database server per service. It is important that the project team analyzes the options and implements the most fitting one. In addition, the database must run in the cloud, meaning there is a strong connection to P3.DevOps within this task. There is a connection to P2 as well, because of the attributes needed for the policies.

(2) Subteam 2 is concerned with developing Consumer-Driven Contract (CDC) tests for CCSApp. CDC tests test the microservices in isolation by using mocks and the contract which defines what is exchanged between the microservices.
(2.1) An important part for the derivation of the contract is the API specification. As a result, a familiarization with the API and the underlying API diagram is necessary. As shown in Figure [1], the excerpt of the CCSApp with the Domain API D-Car and the Process API FleetManagement is considered first. For the desired excerpt, a contract is derived using the engineering artifacts such as the API specification, the constraints, and the task processes.
(2.2) Afterwards, the tests are implemented. For PCM, the framework Pact is used as implementation. One goal of the project team is to find out if this is also possible with the MuleSoft environment.

(3) A further part of subteam 2 is to develop end-to-end tests for applications from the two different engineering approaches. End-to-end tests ensure that the software works as intended from a user’s perspective which means that frontend inputs lead to expected outcomes and the system behaves as intended. Both approaches, C&M’s microservice engineering and the MuleEng approach are tackled by the project team. The application agnostic parts should be carried out in the same way, while the implementation of the end-to-end tests may differ.
(3.1) Both applications use different analysis artifacts. PCM is specified with User/System Interactions (USIs) while CCSApp is specified using use cases. For both analysis artifacts, the derivation of the end-to-end tests is required. For the derivation process, guidelines should be created and followed. A suitable format for specifying end-to-end tests is Gherkin.
(3.2) The test should not be implemented in the user interface directly, but in a separate repository. This increases the reusability of the tests, especially if the user interface is changed (or a second user interface for another device is created).
(3.2.1) One goal is to implement the specified tests in the designated test repository. For each approach, the end-to-end tests are implemented.

<table>
<thead>
<tr>
<th>CCSApp</th>
<th>ConnectedCarServicesApplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDC</td>
<td>Consumer-Driven Contract</td>
</tr>
<tr>
<td>PCM</td>
<td>PredictiveCarMaintenance</td>
</tr>
<tr>
<td>USIs</td>
<td>User/System Interactions</td>
</tr>
</tbody>
</table>
PT Sänger is concerned with the integration of externalized fine-grained authorization mechanisms in the engineering of a microservice-based application. As a policy engine, Open Policy Agent (OPA) is used with its policy language Rego.

1) Subteam 1 researches the derivation of attributes which can be used for Attributed-Based Access Control (ABAC) policies.

1.1) Before ABAC policies can be developed, a set of attributes which can be evaluated must be developed. To ensure a structure-preserving engineering, the attributes must be derived from analysis and design artifacts. Subteam 1 must evaluate an existing approach developed in the last semester and extend it, if necessary.

1.2) During the implementation of authorization policies using OPA, the attributes must be located in the persisted data of the microservice-based application and integrated into the policy. Thus, several options for the integration of attributes (e.g., through HTTP requests) must be evaluated and exemplary implemented.

1.3) Writing an ABAC policy can require to access attributes provided by another microservice or database. Thus, to secure the communication between services, the access must also be authorized requiring service-to-service authorization.

2) Subteam 2 investigates the systematic implementation of authorization policies using OPA. To be able to guarantee the security of a microservice, the policies must also be tested extensively.

2.1) To develop authorization policies in Rego, a systematic procedure must be established. The procedure should take into account the previously created design artifacts.

2.2) Similar to the implementation of microservice source code, the development of authorization policies must be thoughtfully structured and stored in a repository. As a tool for implementing Rego policies, the Styra Declarative Authorization Service (DAS) can be used.

2.3) Currently, the user interface of the ConnectedCarServicesApplication (CCSApp), which is implemented in Angular, uses an implementation of the CanActivate interface to enforce authorization. However, OPA can transform Rego policies into Web Assembly (Wasm) which can be integrated into an Angular user interface. This option should be investigated to ensure a systematic approach to authorization in the user interface.

<table>
<thead>
<tr>
<th>ABAC</th>
<th>Attribute-Based Access Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSApp</td>
<td>ConnectedCarServices.Application</td>
</tr>
<tr>
<td>DAS</td>
<td>Declarative Authorization Service</td>
</tr>
<tr>
<td>OPA</td>
<td>Open Policy Agent</td>
</tr>
<tr>
<td>Wasm</td>
<td>WebAssembly</td>
</tr>
</tbody>
</table>
PT Throner is concerned with the development of a sustainable deployment concept and consideration of development aspects in deployment automation. Helm, Kubernetes, and GitLab CI/CD are used for the implementation.

(1) Subteam 1 is concerned with orchestrating MuleSoft-based applications and developing them into a concept that will enable developers to implement functionality and test the CCSApp in an external system without affecting other developers or systems.

(1.1) Software is developed in increments nowadays. Features are seen as part of the functionality of a software system. To implement a new feature or to adapt an existing one, it is often necessary to adapt several services or to create new ones. In the following the requirements for a smooth and simple development are to be examined.

(1.2) A concept for the provisioning of separate test and development environments is to be established. To reduce costs and minimize start-up time, the environment should only contain the services required for development.

(1.3) The previously developed concept should be integrated into the CCSApp. In addition, a guideline for the provisioning of a feature development environment should be made available to the developers.

(2) Subteam 2 is working on the integration of a test suite into the automated deployment process for Mule applications. A special focus will be on integration and E2E testing of the CCSApp.

(2.1) To identify potential integration test and dependencies of test executions, the relationships between the services must be identified. Logical testable systems should be identified and combined to a testable unit.

(2.2) A fully functional test environment for end-to-end (E2E) test execution requires many resources. To minimize costs and energy consumption, these environments should be shut down after successful test execution.

(2.3) The concept developed for short-lived environments should be implemented and applied to the pipeline concept of the CCSApp. The focus will be on the execution of E2E and integration tests based on the C&M test concept.
Requirements for Participating in the Practical / Seminar Course

(1) Project team meeting each week
   (1) P1.MicroserviceEngineering: Monday, 10 am – 11 am
   (2) P2.AccessPolicies: Monday, 11 am – 12 am
   (3) P3.DevOps: Monday, 12 am – 1 pm

(2) Presentation of the results in the WASA lecture
   (1) See dates for each project team on the page ++Lecture Plan++

(3) 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) 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) Pname, surname, matriculation number, interest in practical course (preferred) or (pro)seminar course
   (2) Personal motivation and experiences in this area (e.g., IT project experiences)
   (3) Priority of the project teams (highest priority first)

(3) 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) The list should contain all project teams. 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 document repository in the following folder:
\sccfs.scc.kit.edu\OE\TM\VR\Mitglieder\2-1.WASA_Aktuell
The C&M document repository is described in detail in the document C&M-TEAMARBEIT (in German) which is available:

(i) on the C&M website: https://cm.tm.kit.edu/

(ii) on the C&M document repository: \sccfs.scc.kit.edu\OE\TM\VR\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.