



A Developer Portal for DevOps Environment

ICSEA 2021 03-07 October 2021 in Barcelona, Spain

COOPERATION & MANAGEMENT (C&M, PROF. ABECK), INSTITUTE OF TELEMATICS, DEPARTMENT OF INFORMATICS

Niklas Sänger Stefan Throner Simon Hanselmann Michael Schneider Sebastian Abeck Contact: niklas.saenger@kit.edu

About Me



- (1) Master's degree in computer science at the Karlsruhe Institute of Technology (KIT)
- (2) Doctoral student at KIT with the following research interests
 - (1) Systematic development of microservices-based applications
 - (2) API management
 - (3) DevOps and cloud technologies



Motivation

- (1) Microservice-based applications are composed of distributed microservices in a cluster
 - (1) Microservices are typically developed by multiple development teams
 - (2) APIs can act as a contract between microservices
- (2) Providing information about a microservice is crucial for the development
 - (1) Manual updating of documentation can lead to data inconsistencies
 - (2) Frequent changing developers and distributed teams can degrade the knowledge about microservices
- (3) Support of developers by providing a point of contact offering information about microservices
 - (1) Automated solution are used to provide source-of-truth

Cooperation & Management (C&M, Prof. Abeck) Institute of Telematics, KIT Department of Informatics

Goal of the MicroserviceDeveloperPortal (MDP)

Develop a microservice-based application (1)

(2)

- Following a systematic microservice engineering approach (1)
- Support developers by providing necessary information of deployed microservices Automated data collection (1) Repository (e.g., API, URL) using a built and CI/CD stage Meta deployed to monitors organized in Documentation Data Health monitoring (2) Monitorina (running) GitLab Microservices provides interacts with Specification Dependencies (re)uses requests Health Two capabilities (3) **Provision of CompileTimeData** Provision of RuntimeData requests Provision of RuntimeData (1) Developer Provision of CompileTimeData (2)



The MDP Dashboard



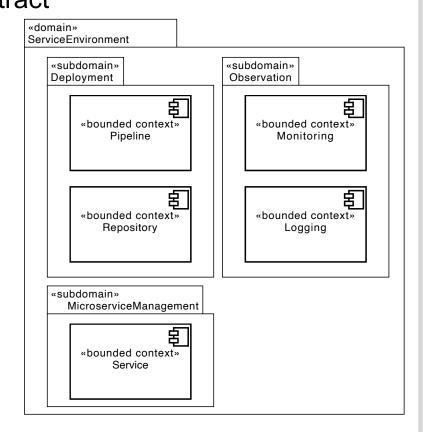
- (1) Dashboards provides a list of registered microservices
- (2) Selected service provides more detailed information and actions
 - (1) Hostname, health, version, address, code owner, Helm dependencies, links, and Kubernetes Pods

\equiv MDP Dashboard				\equiv MDP Dashboard					
List of Microservices			last refreshed: 15:39:14 GMT+0200 (CEST)	∠ Back cm-api-management (<i>master</i>)					
Hostname	Health		Version Address	Actions:	Open Delete				
cm-api-management	UP	STABLE	master cm-api-management.cloud.iai.kit.edu	Health:		ABLE			
cm-medicaldevice	× DOWN		master cm-medicaldevice.cloud.iai.kit.edu	Code Owner:	@Vorname.Name @	Vorname2.Nar	me2		
cm-vehicle	UP	STABLE	master cm-vehicle.cloud.iai.kit.edu	Dependencies:					
				Dependencies.	template-chart: - ~1.0.0				
				Links:	REPOSITORY	PIPELINE	API	SWAGGER	
				Deployed Pods:	Name			ReplicaSet	Phase
					cm-api-management-56f45dcf65-hqhbj cm-api-management-56f45dcf65 Running				

(1) New domain to place the MDP and extract application-agnostic logic

- (1) GitLab as bounded context Repository
- (2) GitLab CI/CD pipeline as bounded context Pipeline
- (2) Bounded contexts implemented as domain microservices
 - (1) Service stores microservice information (e.g., version)
 - (2) Monitoring stores health data (e.g., up, down)
- (3) Domain microservices provide Create, Read, Update, and Delete (CRUD) operations

Domain ServiceEnvironment





Application Sharing View



- (1) Bounded contexts persist and share entities
 - (1) Repository
 - (2) Microservice
 - (3) Health

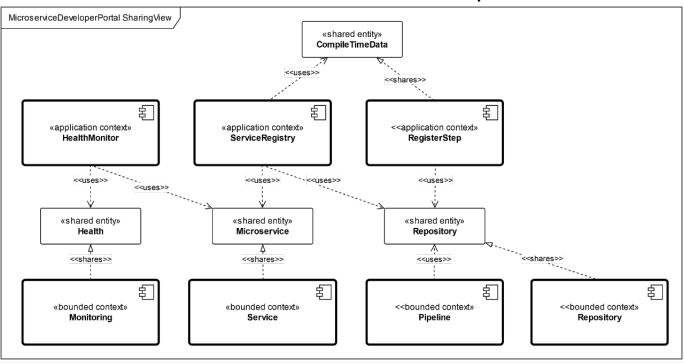
7

(2) Application contexts create shared entities

CompileTimeData

- (1) Microservice
- (2) Health

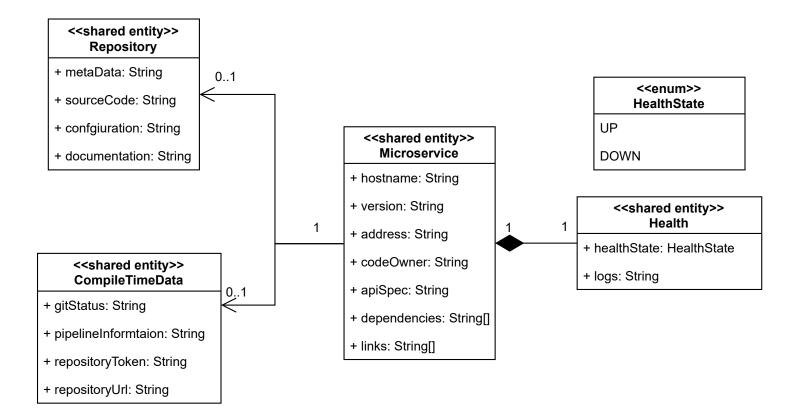
(3)



Relation View



(1) The relation view captures the entities shared by the bounded contexts and sets them into relation



Niklas Sänger et al.: A Developer Portal for DevOps Environment 9 17.09.21

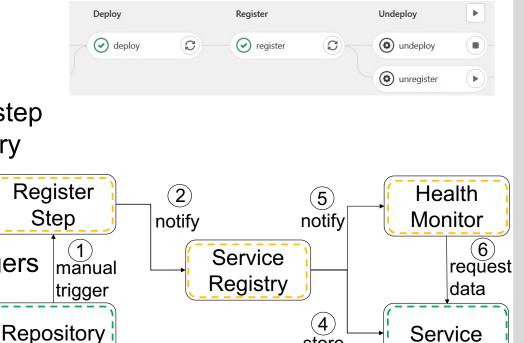
Cooperation & Management (C&M, Prof. Abeck) Institute of Telematics, KIT Department of Informatics

Application Context Bounded Context

store

Microservice Registration Process

- Two stages added to a CI/CD (1) pipeline (i.e., register, unregister)
 - Docker image executing (1) Python scripts
- Registration of a microservice (2) triggered by a commit
 - Executed after the deploy step (1) notifying the ServiceRegistry
- The ServiceRegistry stores (3) a new entity in domain microservice Service and triggers HealthMonitor
- Unregistration in parallel to (4) undeploy stage





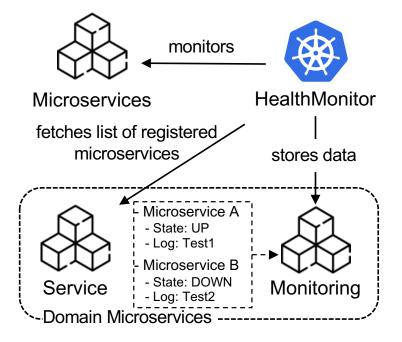
Karlsruhe Institute of Technology

(1) Implemented as a Kubernetes

Health Monitor

operator

- (1) Monitoring of Kubernetes Pod lifecycle (e.g., running, terminated)
- (2) Changes in pod lifecycle trigger an event
- (2) HealthMonitor fetches a list of registered microservices from the domain microservice Service
- (3) Health data is stored in the domain microservice Monitoring



Results and Outlook



- (1) A reusable microservice-based application called MicroserviceDeveloperPortal to support developers
 - (1) Providing CompileTimeData and RuntimeData to see the information about deployed microservice
 - (2) Utilizing a CI/CD pipeline to automate data collection
- (2) A domain ServiceEnvironment representing an environment in which microservices are developed and operated
- (3) Extend the MDP with a focus on the management of APIs
 - (1) Replace the HealthMonitoring mechanism
- (4) Remodeling of the RegisterStep to be part of the domain ServiceEnvironment



Thank You for Your Attention