Towards Semi-Automated Viewpoint-based Reconstruction of Microservice Architecture

AMP 2022, Copenhagen, Denmark

Philip Wizenty

philip.wizenty@fh-dortmund.de

13 Jun 2022

University of Applied Sciences and Arts Dortmund,
IDiAL Institute

Inhaltsverzeichnis

Table of Content

Motivation

- MSA promotes to increase service independence by
  - letting it realize a distinct, self-contained capability
  - decreasing its coupling to other software components w.r.t., e.g., implementation, testing, and operation
  - transferring its ownership to a dedicated team, being responsible for all aspects related to service design, implementation, and operation
  - add modifiability
- Improved maintainability by facilitating the replacement of services with improved versions

Motivation

- MSA promotes to increase service independence by
  - letting it realize a distinct, self-contained capability
  - decreasing its coupling to other software components w.r.t., e.g., implementation, testing, and operation
  - transferring its ownership to a dedicated team, being responsible for all aspects related to service design, implementation, and operation
  - add modifiability
- Improved maintainability by facilitating the replacement of services with improved versions

Motivation

- MSA promotes to increase service independence by
  - letting it realize a distinct, self-contained capability
  - decreasing its coupling to other software components w.r.t., e.g., implementation, testing, and operation
  - transferring its ownership to a dedicated team, being responsible for all aspects related to service design, implementation, and operation
  - add modifiability
- Improved maintainability by facilitating the replacement of services with improved versions

Motivation

Problem Statement

- Increased modifiability facilitates service evolution
- Increased independence enables teams to autonomously adapt different parts of the software system

$\Rightarrow$ Increased risk for the erosion of the anticipated architecture design

Motivation

Solution Proposal

- Software Architecture Reconstruction (SAR) [1] to (semi-) automatically recover a microservice architecture's design
- Model-based SAR to recover architecture information from different viewpoints
- The viewpoints addressing concerns of different type of stakeholders in the software engineering process
- Models to facilitate the engineering process of the MSA-based software system

Table of Content

Viewpoint-based MSA Modeling with LEMMA

- Model-driven Engineering (MDE) [2] is an approach to software engineering that aims to facilitate the design, implementation, and operation of a software system though the use of *models*
- A *model* [2] in sense of MDE is an artifact that:
  - Abstracts from selected characteristics of the considered software system
  - Is expressed in a dedicated modeling language
  - Is (semi-) automatically processible for specific purposes in the software engineering process

Viewpoint-based MSA Modeling with LEMMA

- Model-driven Engineering (MDE) [2] is an approach to software engineering that aims to facilitate the design, implementation, and operation of a software system though the use of *models*
- A *model* [2] in sense of MDE is an artifact that:
  - Abstracts from selected characteristics of the considered software system
  - Is expressed in a dedicated modeling language
  - Is (semi-) automatically processible for specific purposes in the software engineering process

Viewpoint-based MSA Modeling with LEMMA

- Model-based *viewpoints* [4, 3] provide means to reduce the software system's complexity by describing only a specific part of the system
- *View models* are specifically effective in making the parts and underlying concepts of complex software architectures explicit to facilitate the reasoning about them [6]

Viewpoint-based MSA Modeling with LEMMA

- Model-based *viewpoints* [4, 3] provide means to reduce the software system's complexity by describing only a specific part of the system
- *View models* are specifically effective in making the parts and underlying concepts of complex software architectures explicit to facilitate the reasoning about them [6]

Viewpoint-based MSA Modeling with LEMMA

- LEMMA[1] is an MDE-based ecosystem that focuses on the concerns of different stakeholder groups in MSA engineering
- LEMMA enables the construction of models for. . .
  - . . . domain-driven service design (Domain Data Modeling Language)
  - . . . API management (Service Modeling Language)
  - . . . service operation (Operation Modeling Language)

[1] `https://fh.do/lemma`

Viewpoint-based MSA Modeling with LEMMA

- LEMMA[1] is an MDE-based ecosystem that focuses on the concerns of different stakeholder groups in MSA engineering
- LEMMA enables the construction of models for...
  - ... domain-driven service design (Domain Data Modeling Language)
  - ... API management (Service Modeling Language)
  - ... service operation (Operation Modeling Language)

[1] `https://fh.do/lemma`

Table of Content

LEMMA-Enabled Approach for MSA Reconstruction

- (a) LEMMA-Enabled Microservice Architecture Reconstruction (MAR) Framework
  - Orchestrates the stages of the SAR process
  - Provides functionalities for reconstructing viewpoint-specific information
  - Manages MAR plugins
- (b) MAR Plugins
  - Derive viewpoint-specific architecture information from source code artifacts

LEMMA-Enabled Approach for MSA Reconstruction

- (a) LEMMA-Enabled Microservice Architecture Reconstruction (MAR) Framework
  - Orchestrates the stages of the SAR process
  - Provides functionalities for reconstructing viewpoint-specific information
  - Manages MAR plugins
- (b) MAR Plugins
  - Derive viewpoint-specific architecture information from source code artifacts

LEMMA-Enabled Approach for MSA Reconstruction

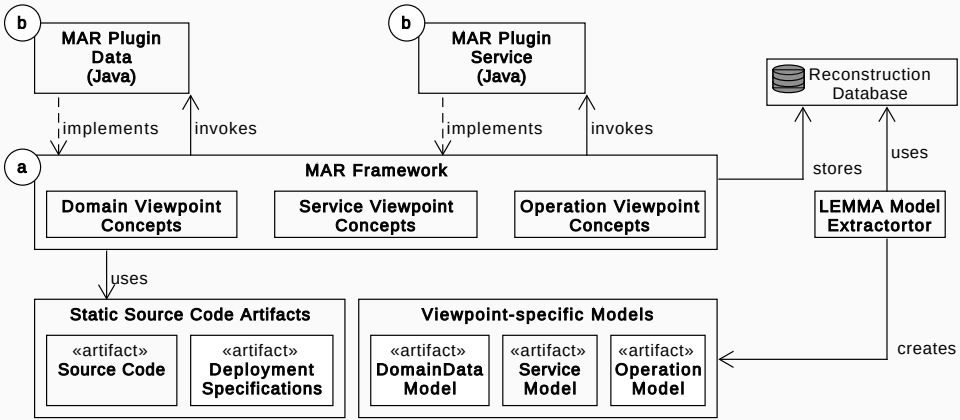

Figure 1: LEMMA-Enabled Approach for MSA Reconstruction.

LEMMA-Enabled Approach for MSA Reconstruction

Listing 1: Example Java source code artifact.

```
1  @Entity
2  @Table(name = "interactions")
3  public class InteractionEntity {
4    @Id
5    private String id;
6    private Date date;
7    private String content;
8    private boolean sentByOperator;
9    ...
10 }
```

Listing 2: Reconstructed LEMMA domain model.

```
1  context customerManagementBackend {
2    structure InteractionEntity<entity> {
3      string id<identifier>,
4      date createDate,
5      string content,
6      boolean sentByOperator,
7      ...
8    }
9  }
```

LEMMA-Enabled Approach for MSA Reconstruction - One Slide

Listing 3: Example Java source code artifact.

```
1 @RestController
2 @RequestMapping("/customers")
3 public class CustomerInformationHolder {
4   @GetMapping(value = "/{customerId}")
5   public ResponseEntity<CustomerDto> getCustomer(
6     @PathVariable CustomerId customerId) {...
7     return ResponseEntity.ok(customer);}}
```

Listing 4: Reconstructed LEMMA service model.

```
1   public functional microservice com.lakesidemutual.CustomerManagement {
2     interface CustomerInformationHolder {
3       getCustomers(
4         sync out customer : Customer::Customer.PaginatedCustomerResponseDto,
5         sync in filter : string,  sync in integer : customerId);
6 ...}}
```

Table of Content

Validation of LEMMA's Reconstruction Framework

- *Binary Classification* [5] for the validation of the reconstruction results
- Classification of the reconstruction architecture information:
  - *True positive* (TP): Correctly reconstructed
  - *True negative* (TN): Not reconstructed
  - *False positive* (FP) / *False negative* (FN): Wrongly reconstructed

Validation of LEMMA's Reconstruction Framework

- *Binary Classification* [5] for the validation of the reconstruction results
- Classification of the reconstruction architecture information:
  - *True positive* (TP): Correctly reconstructed
  - *True negative* (TN): Not reconstructed
  - *False positive* (FP) / *False negative* (FN): Wrongly reconstructed

Validation of LEMMA's Reconstruction Framework

1. *Recall* [5]: Probability to identify a relevant element

$$Recall = \frac{TP}{TP + FN} \tag{1}$$

2. *Precision* [5]: The correctness of the reconstructed elements

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

3. $F_{measure}$ [5]: Accuracy of the entire reconstructed architectural design

$$F_{measure} = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{3}$$

Validation of LEMMA's Reconstruction Framework

1. *Recall* [5]: Probability to identify a relevant element

$$Recall = \frac{TP}{TP + FN} \tag{1}$$

2. *Precision* [5]: The correctness of the reconstructed elements

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

3. $F_{measure}$ [5]: Accuracy of the entire reconstructed architectural design

$$F_{measure} = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{3}$$

Validation of LEMMA's Reconstruction Framework

1. *Recall* [5]: Probability to identify a relevant element

$$Recall = \frac{TP}{TP + FN} \tag{1}$$

2. *Precision* [5]: The correctness of the reconstructed elements

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

3. $F_{measure}$ [5]: Accuracy of the entire reconstructed architectural design

$$F_{measure} = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{3}$$

Validation of LEMMA's Reconstruction Framework

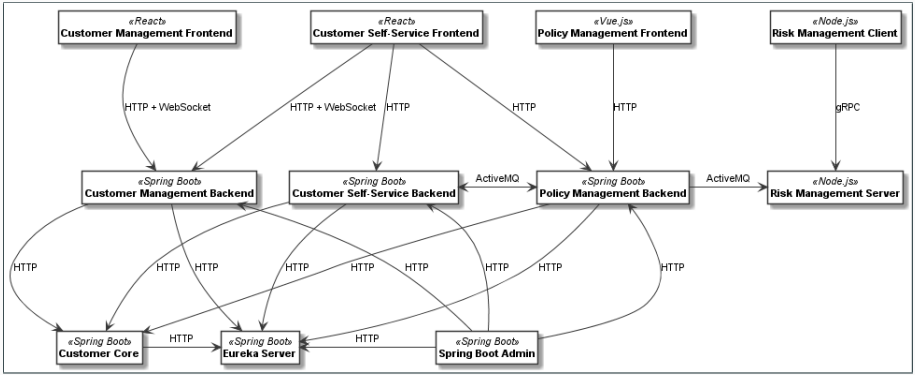

Figure 2: Intended architecture design of Lakeside Mutual[2].

[2] https://github.com/Microservice-API-Patterns/LakesideMutual

Validation of LEMMA's Reconstruction Framework

Table 1: Results for the reconstruction of the architecture design of Lakeside Mutual[3].

Java | Domain Data Model | Service Model

Element	Expected	TP	FP	FN	Recall	Precision	$F_{measure}$
Microservices	5	4	0	1	80%	100%	88%
Interfaces	16	14	0	2	87%	100%	93%
Operations	61	50	3	8	86%	94%	90%
Data Structures	161	117	29	14	89%	80%	84%

[3] https://github.com/SeelabFhdo/microservices2022

Questions

Questions?

Literatur

[1] Len Bass, Paul Clements, and Rick Kazman. Software architecture in practice. Addison-Wesley Professional, 2003.

[2] Benoit Combemale et al. Engineering modeling languages: Turning domain knowledge into tools. CRC Press, 2016.

[3] Robert France and Bernhard Rumpe. “Model-driven development of complex software: A research roadmap.” In: Future of Software Engineering (FOSE’07). IEEE. 2007, pp. 37–54.

[4] ISO/IEC/IEEE. Systems and software engineering — Architecture description. Standard ISO/IEC/IEEE 42010:2011(E). 2011.

Literatur

[5] Robert Stahlbock. Advances in Data Science and Information Engineering. Ed. by Gary M. WeissMahmoud Abou-NasrCheng-Ying YangHamid R. ArabniaLeonidas Deligiannidis. Springer, Cham, 2021.

[6] Jon Whittle, John Hutchinson, and Mark Rouncefield. “The State of Practice in Model-Driven Engineering.” In: IEEE Software 31.3 (May 2014). IEEE, pp. 79–85.