

Test Makroarchitektur, Teil 2

(Schwerpunkt der Überprüfung sind Aufgaben 6-10 des Übungszettels, also eigenständiges Entwerfen/Codieren)

Aufgabenstellung "Buchhandlung"

Es soll der Use Case einer **Buchbestellung** umgesetzt werden:

1. es wird eine Buchbestellung angenommen
2. danach wird der **Buchdruck passend zur Bestellung durchgeführt**
3. sobald der Buchdruck abgeschlossen ist, kann **das Buch "abgeholt"** werden.

Bitte in der folgenden Detailbeschreibung die Namen in Anführungszeichen auch genau so im Programm benennen!

Details:

- Über eine REST-API kann beim "**buchhandlung-service**" die Buchbestellung initiiert werden. Dazu wird ein entsprechendes **"Bestellung"-Objekt** angelegt und in der Datenbank gespeichert.
- Das "**buchhandlung-service**" übermittelt in weiterer Folge ein **"BuchBestelltEvent"** (dieses Event hat genau diesen Namen) an den Message Broker (idealerweise RabbitMQ Message Broker, wie bereits in Aufgabe 6 "Cargo" verwendet -> gern dort nachsehen; zur Erinnerung: auf Moodle findet ihr auch das Video zur Inbetriebnahme von Cargo mit RabbitMQ). **Nach erfolgreichem** Versand des Event wird am zugehörigen **"Bestellung"-Objekt ein Status "BESTELLT" gesetzt**.
- Das **"druckerei-service"** **reagiert auf das "BuchBestelltEvent"** (erstellt und sendet vom "**buchhandlung-service**") und **führt den Buchdruck durch**.
- **Sobald** der Buchdruck im "**druckerei-service**" durchgeführt wurde, übermittelt das "**druckerei-service**" das **"BuchGedrucktEvent"** über den Message Broker. Über den Message Broker wird dieser Event an entsprechende Konsumenten (in diesem Fall lediglich das "**buchhandlung-service**") weiter geleitet.
- Bei Erhalt des "**BuchGedrucktEvent**" setzt der "**buchhandlung-service**" den Status am zugehörigen "**Bestellung"-Objekt** auf **"ABHOLBEREIT"**

Zusätzliche Rahmenbedingungen:

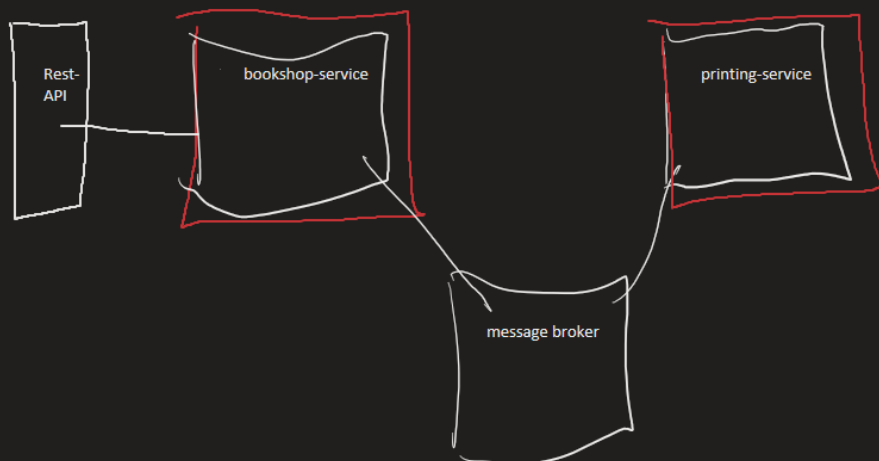
- zuerst ist die Architektur zu entwerfen; dazu sind geeignete Diagramme bzw. Handskizzen zu erstellen

Use case: Buchbestellung

Buchhandlung-service

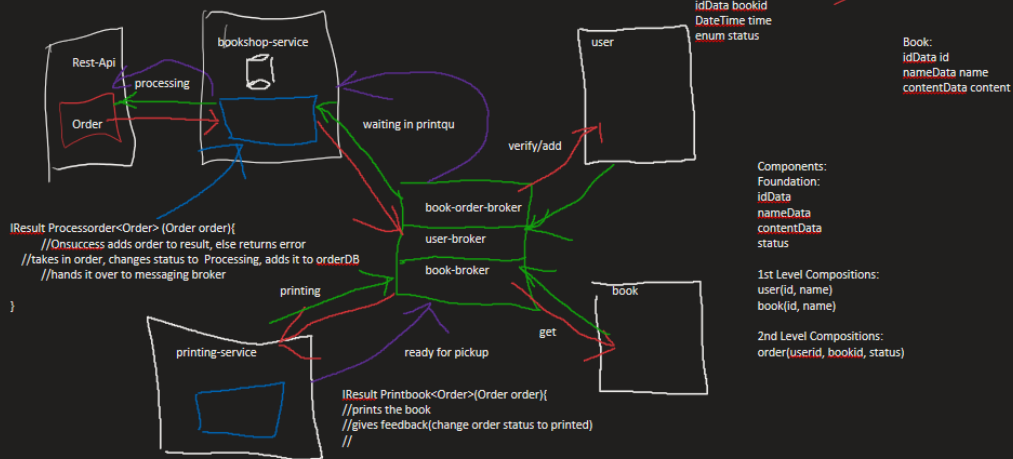
Microservices

Wednesday, 28 April 2021 14:03



Order Use-Case

Wednesday, 28 April 2021 14:08



Input => Process => Output

username, bookname, DateTime	Fetch	Book, User, DateTime
Book, User, DateTime	Verify	Result (Order without ID)
Order	PlaceOrder	Order => also places order in OrderDB
Order	PrintbookCommand	PrintCommand(Order)
PrintCommand()	FetchBookContents	Book
Order, bookcontent	Print	Order

green arrow callback
 red arrow command
 purple arrow finished

```

Project
  bookms D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\bookms
  bookstore D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\bookstore
  domain D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\domain
    .mvn
    .settings
    src
      main
        java
          at
            itkollegimst
              hampl
                pos1makro
                  test2
                    domain
                      common
                        abstracts
                          ICommand.java
                          IResult.java
                        entities
                          Book.java
                          Order.java
                          User.java
                      DomainApplication.java
  
```

```

BookstoreApplication.java
IResult.java
1 package at.itkollegimst.hampl.pos1makro.test2.domain.domain.common.abstracts;
2
3 public abstract class IResult<TData> {
4     private TData result;
5     private Error error;
6
7     public IResult(TData data) { this.result = data; }
8     public IResult(Error error) { this.error = error; }
9
10    public TData GetData() { return result; }
11    public Error GetError() { return error; }
12
13    public boolean isSuccessful() { return result != null; }
14    public boolean failed() { return error != null; }
15 }
  
```

```

Project
  bookms D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\bookms
  bookstore D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\bookstore
  domain D:\itkolleg\ITKolleg.Pos-Ase.Macroarch-Test\domain
    .mvn
    .settings
    src
      main
        java
          at
            itkollegimst
              hampl
                pos1makro
                  buchhandlung
                    bookstore
                      aggregate
                        RESTOrderAggregate.java
                      application
                      repository
                      REST
                    BookstoreApplication.java
  
```

```

BookstoreApplication.java
IResult.java
RESTOrderAggregate.java
1 package at.itkollegimst.hampl.pos1makro.test2.buchhandlung.bookstore.aggregate;
2
3
4 public class RESTOrderAggregate {
5     private String bookName;
6     private String userName;
7     private String userMail;
8
9
10    public RESTOrderAggregate(String bookName, String userName, String userMail) {
11        this.bookName = bookName;
12        this.userName = userName;
13        this.userMail = userMail;
14    }
15
16    public long getOrderid() {
17        return orderId;
18    }
19
20    public void setId(long orderId) {
  
```