Tampere University

General topics

Interesting pilot course (In Finnish)

- Public ICT tenders and provisioning
  (COMP.530-04 Julkiset ICT-hankinnat
  Jos haluat ymmärtää ICT-hankintoja, ilmoittaudu ensi kertaa järjestettävälle ja upouudelle ”Julkiset ICT-hankinnat” kurssille!)
- kurssin tavoitteena on oppia miten suunnitella, kilpailuttaa ja arvioida ICT-hankintoja, kuten tietojärjestelmät ja erilaiset asiantuntijapalvelut. Opit pelisääntöjä ja parhaita käytäntöjä.
- Kurssi tarjoaa myös ainutlaatuisen tilaisuuden verkostoitua. Kurssi suoritetaan pienryhmissä projektityönä, jossa mm. analysoitte yhdessä todellisia hankintoja. Opitte myös toisiltanne ja rakennatte samalla kontakteja tuleviin mahdollisuuksiin. Kurssille voidaan ottaa enintään 30 aktiivista oppilasta, joten kannattaa ilmoittautua heti, jos teema kiinnostaa.
- In SISU: COMP.530-04: Julkiset ICT-hankinnat

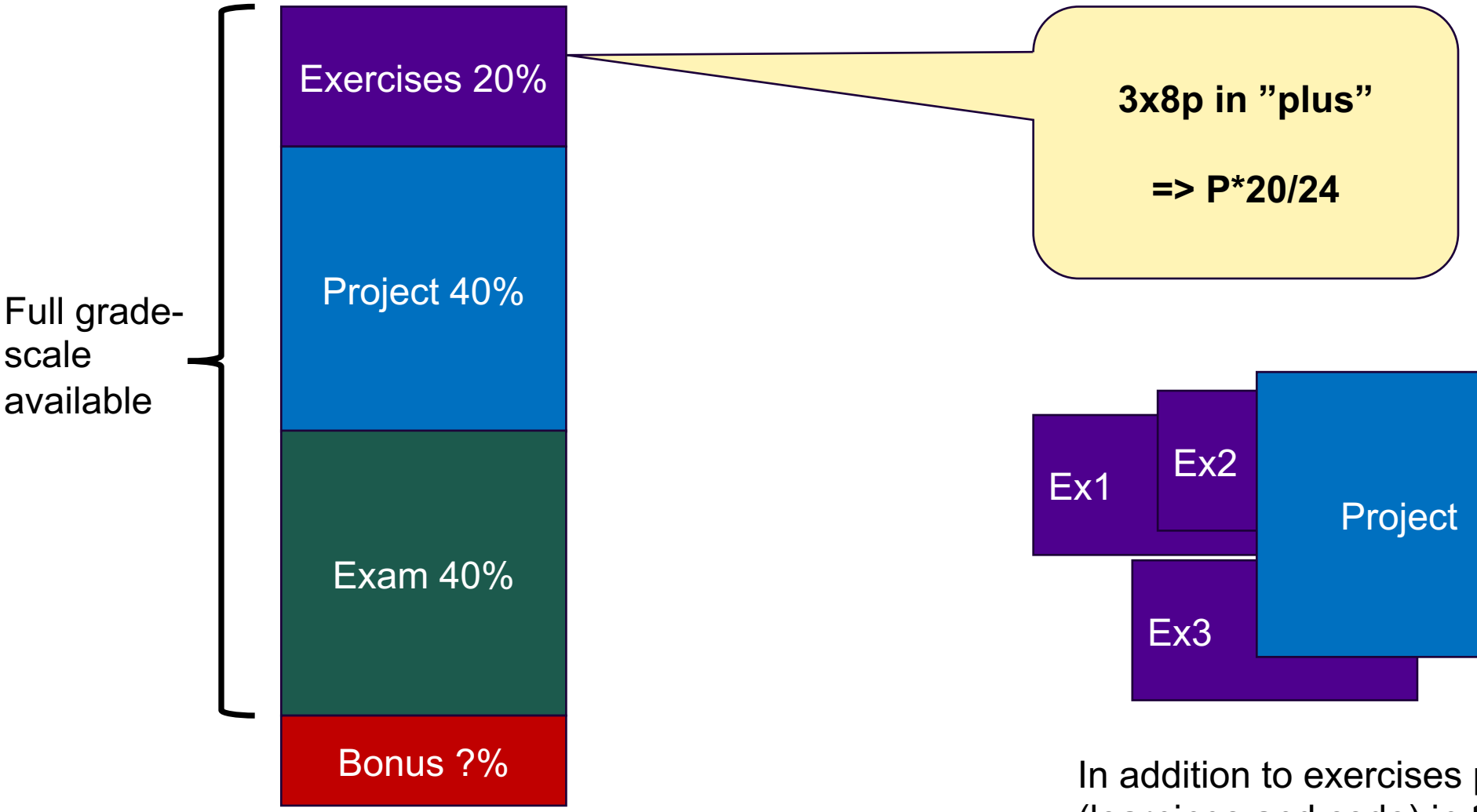
Tampere University

Course status

- 79 submission for the first status
- No f2f session during next two weeks
- Collection of material bank has been started
- Next exercise will be annouced today

Tampere University

On exercises, project, grading and points



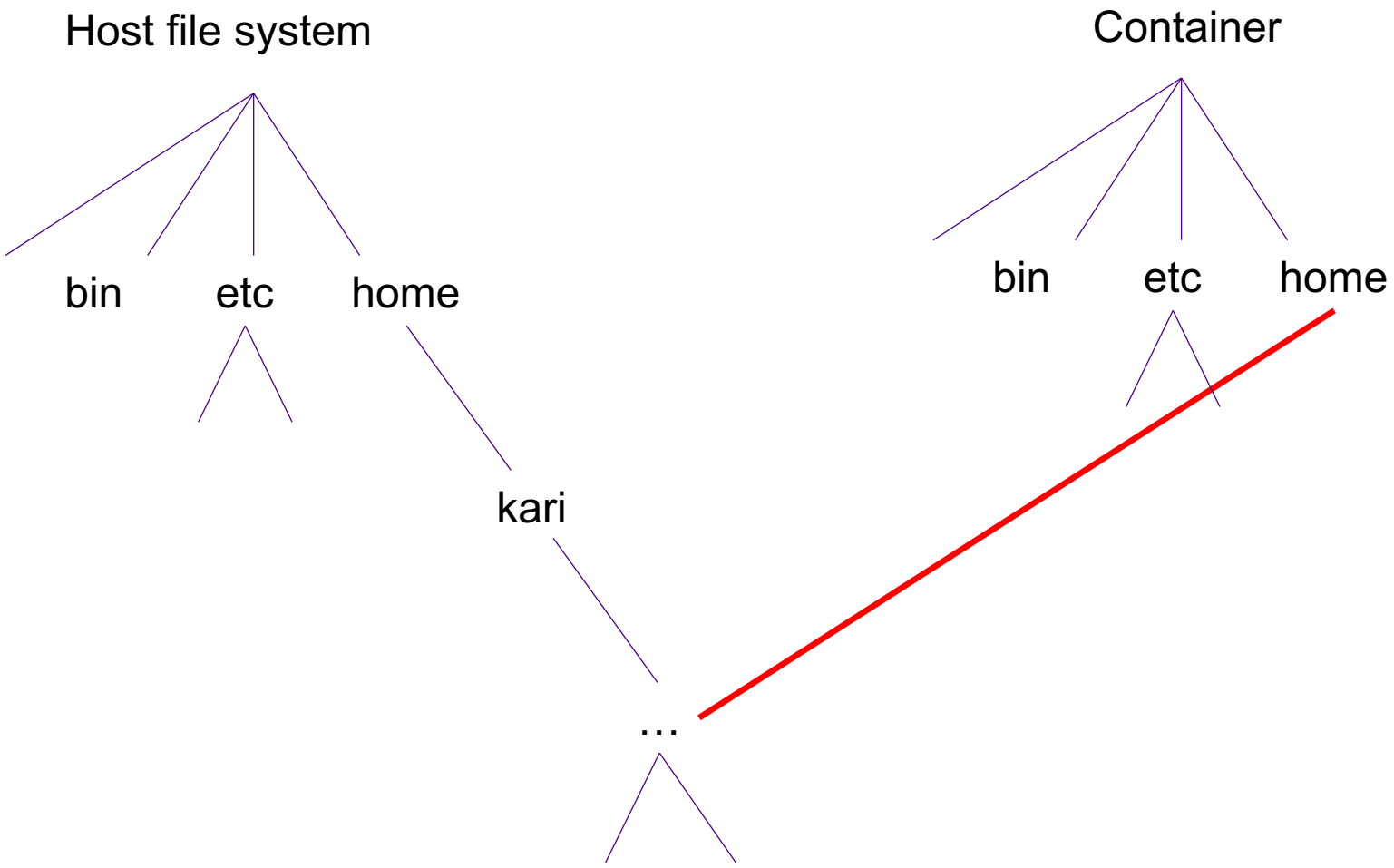
In addition to exercises point you also reuse (learnings and code) in the project

On the previous exercise

What was difficult?
What was stupid?

Tampere University

About volumes

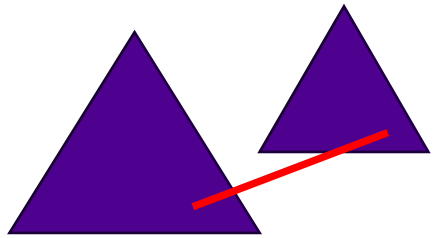


```
FROM ubuntu
WORKDIR home
RUN mkdir /home/test42
CMD [ "sh", "test.sh" ]
```

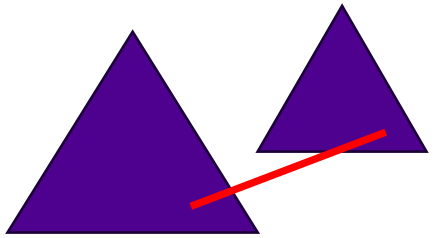
```
services:
  test:
    image: test
    volumes:
    - .:/home
```

```
whoami > test.txt
ls -ls
```

```
Recreating volumetest_test_1 ... done
Attaching to volumetest_test_1
test_1 | total 24
test_1 | 4 -rw-rw-r-- 1 1000 1000 72 Oct 1 14:18 Dockerfile
test_1 | 4 -rw-rw-r-- 1 1000 1000 82 Oct 1 14:15 Dockerfile~
test_1 | 4 -rw-rw-r-- 1 1000 1000 63 Oct 1 14:08 docker-
compose.yaml
test_1 | 4 -rw-rw-r-- 1 1000 1000 62 Oct 1 14:05 docker-
compose.yaml~
test_1 | 4 -rwxrwxr-x 1 1000 1000 25 Oct 1 14:17 test.sh
test_1 | 4 -rw-r--r-- 1 root root 5 Oct 1 14:18 test.txt
volumetest_test_1 exited with code 0
kari@kari-Aspire-5552:~/ComposeTest/VolumeTest$ ls -ls
total 24
4 -rw-rw-r-- 1 kari kari 63 loka 1 17:08 docker-compose.yaml
4 -rw-rw-r-- 1 kari kari 62 loka 1 17:05 docker-compose.yaml~
4 -rw-rw-r-- 1 kari kari 72 loka 1 17:18 Dockerfile
4 -rw-rw-r-- 1 kari kari 82 loka 1 17:15 Dockerfile~
4 -rwxrwxr-x 1 kari kari 25 loka 1 17:17 test.sh
4 -rw-r--r-- 1 root root  5 loka 1 17:18 test.txt
```



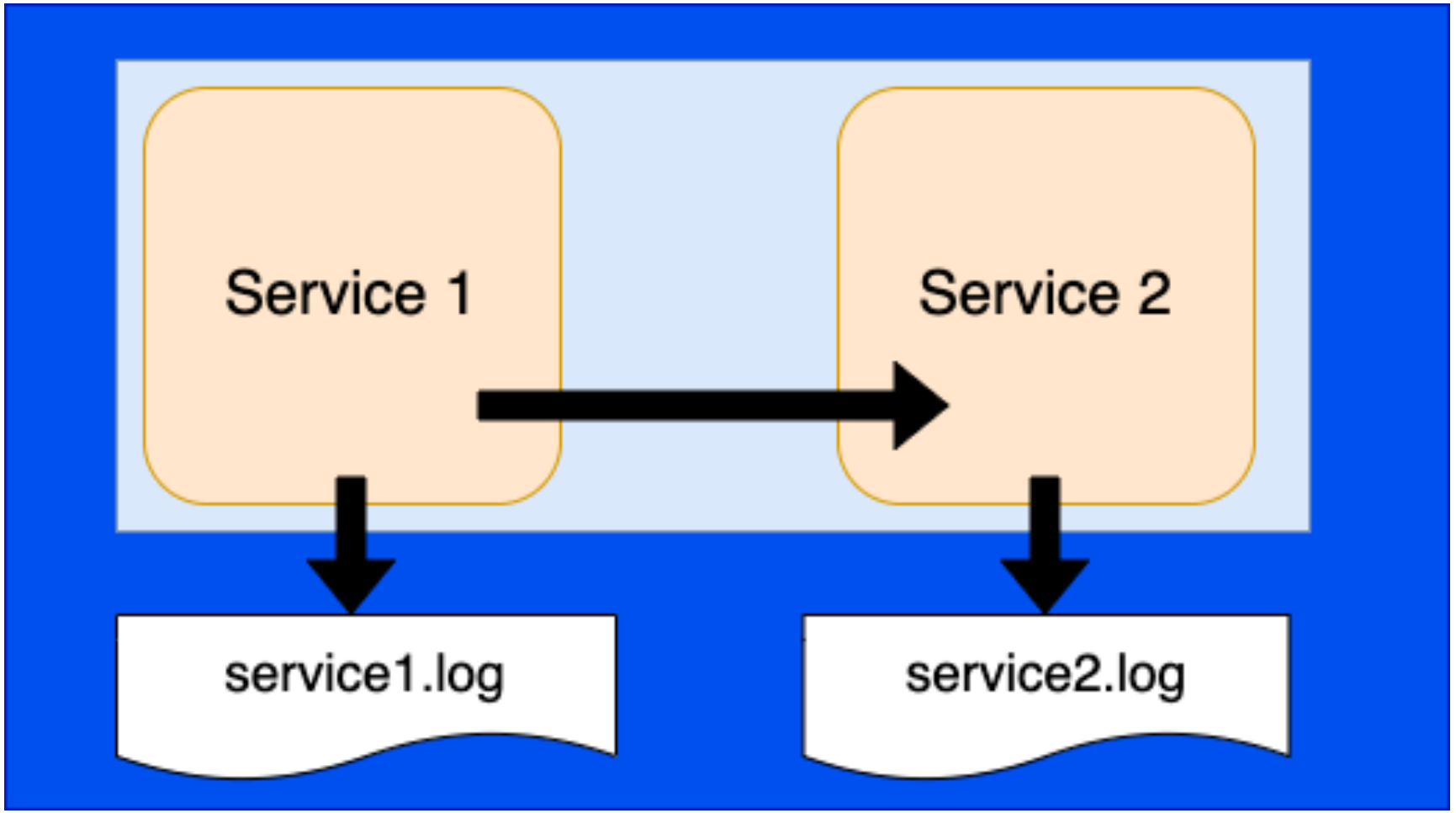
```
FROM ubuntu
RUN mkdir /home/test42
RUN groupadd -r kari && useradd -r -g kari kari
USER kari
WORKDIR home
CMD [ "sh", "test.sh" ]
```



```
sudo docker-compose up
Removing volumetest_test_1
Recreating c56d1d5faf34_volumetest_test_1 ... done
Attaching to volumetest_test_1
test_1  | test.sh: 1: cannot create test.txt: Permission denied
test_1  | total 24
test_1  | 4 -rw-rw-r-- 1 1000 1000 130 Oct  1 14:53 Dockerfile
test_1  | 4 -rw-rw-r-- 1 1000 1000 120 Oct  1 14:52 Dockerfile~
test_1  | 4 -rw-rw-r-- 1 1000 1000  63 Oct  1 14:08 docker-compose.yaml
test_1  | 4 -rw-rw-r-- 1 1000 1000  62 Oct  1 14:05 docker-compose.yaml~
test_1  | 4 -rwxrwxr-x 1 1000 1000  25 Oct  1 14:17 test.sh
test_1  | 4 -rw-r--r-- 1 root root   5 Oct  1 14:18 test.txt
volumetest_test_1 exited with code 0
```

Discussion: what are other problems with volumes

Communication architecture and cloud native principles?



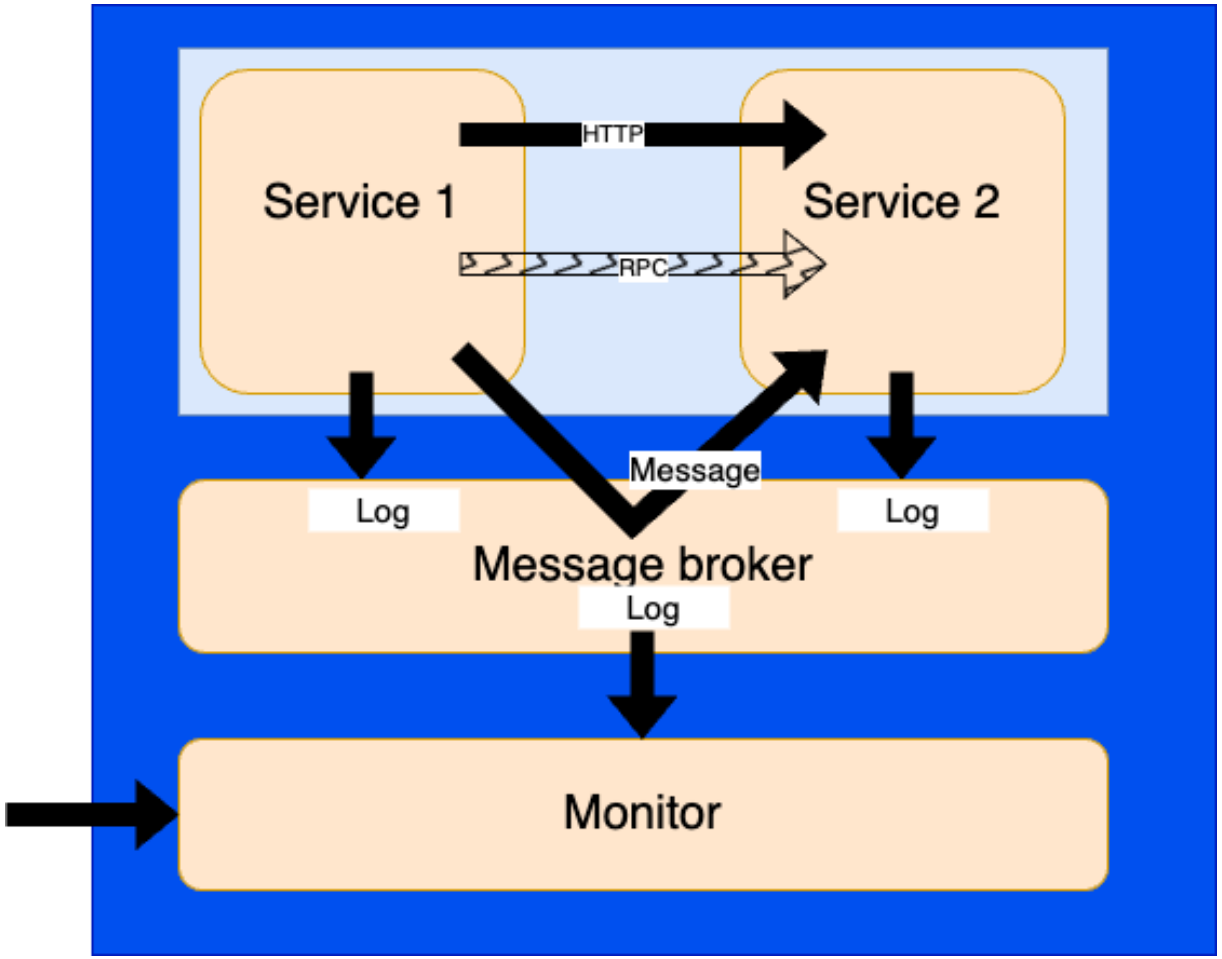
Discussion about communication

When and when not

REST	RPC
GraphQL	MessageQueue/BUS

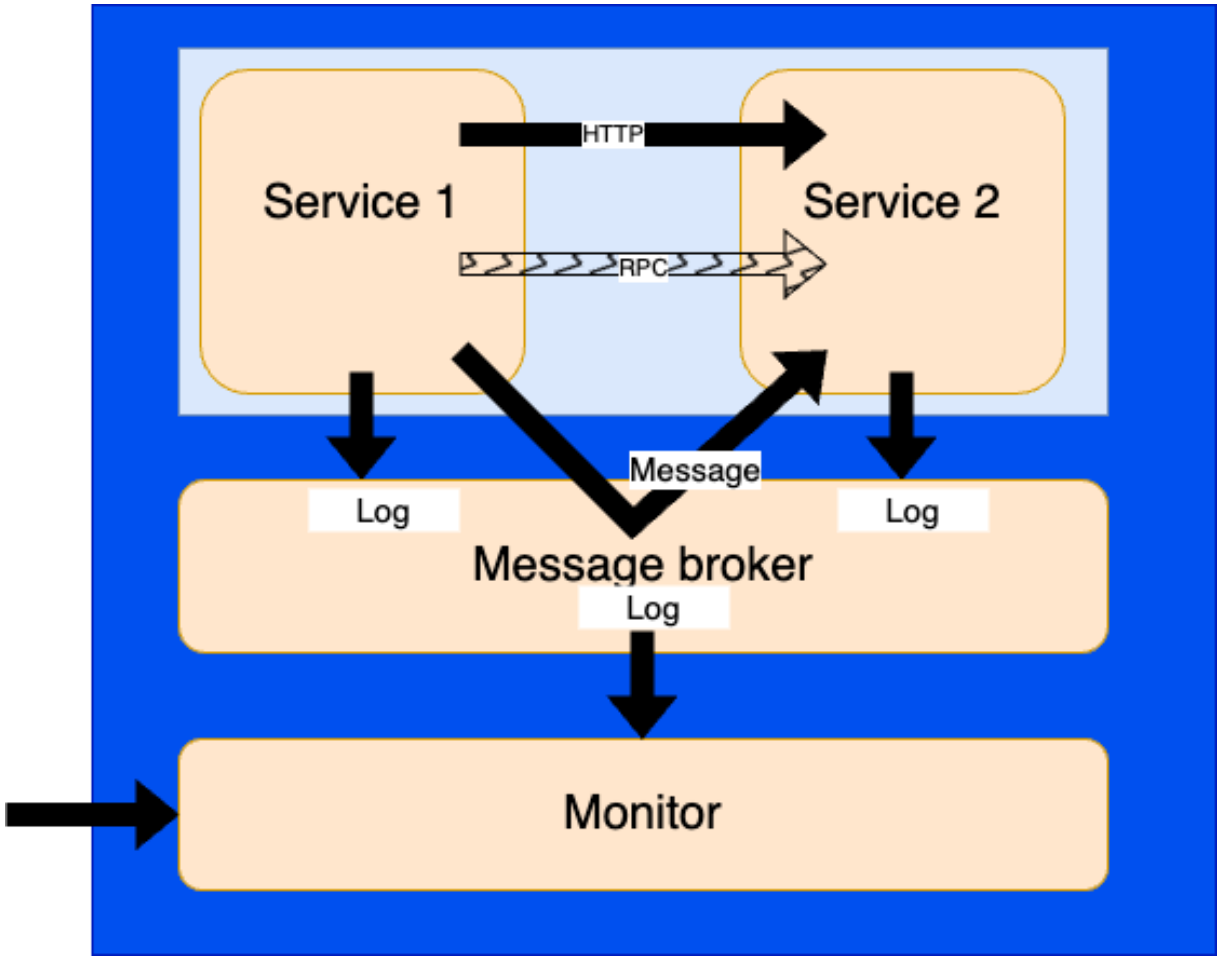
Next exercise

Function of Service 1 (repeated 20 times)



- Compose a **text** from a counter (initialized to 1) current time and address+port of service2.
  `SND 1 2022-10-01T06:35:01.373Z 192.168.2.22:8000`
- Send the above text to message broker (RabbitMQ - topic "message") for Service2
- Send the text with HTTP protocol to service 2. (i.e., the same text is sent through another channel)
- Send the response code with a time stamp of the above topic "log" . Example string:
  `200 2022-10-01T06:35:01.973Z`
- *Optional: Send an RPC (gRPC recommended) request to service 2. Parameter of the called service is the string that was sent via RabbitMQ and HTTP.*
- *Optional: after the call returns the following string is formed*
  `RPC X`
  where "X" is the returned value. The formed string is sent to message broker (topic "log").
- If either sending fails, catch the exception and send the error message to Message Broker (topic "log")
- Increase the counter with 1

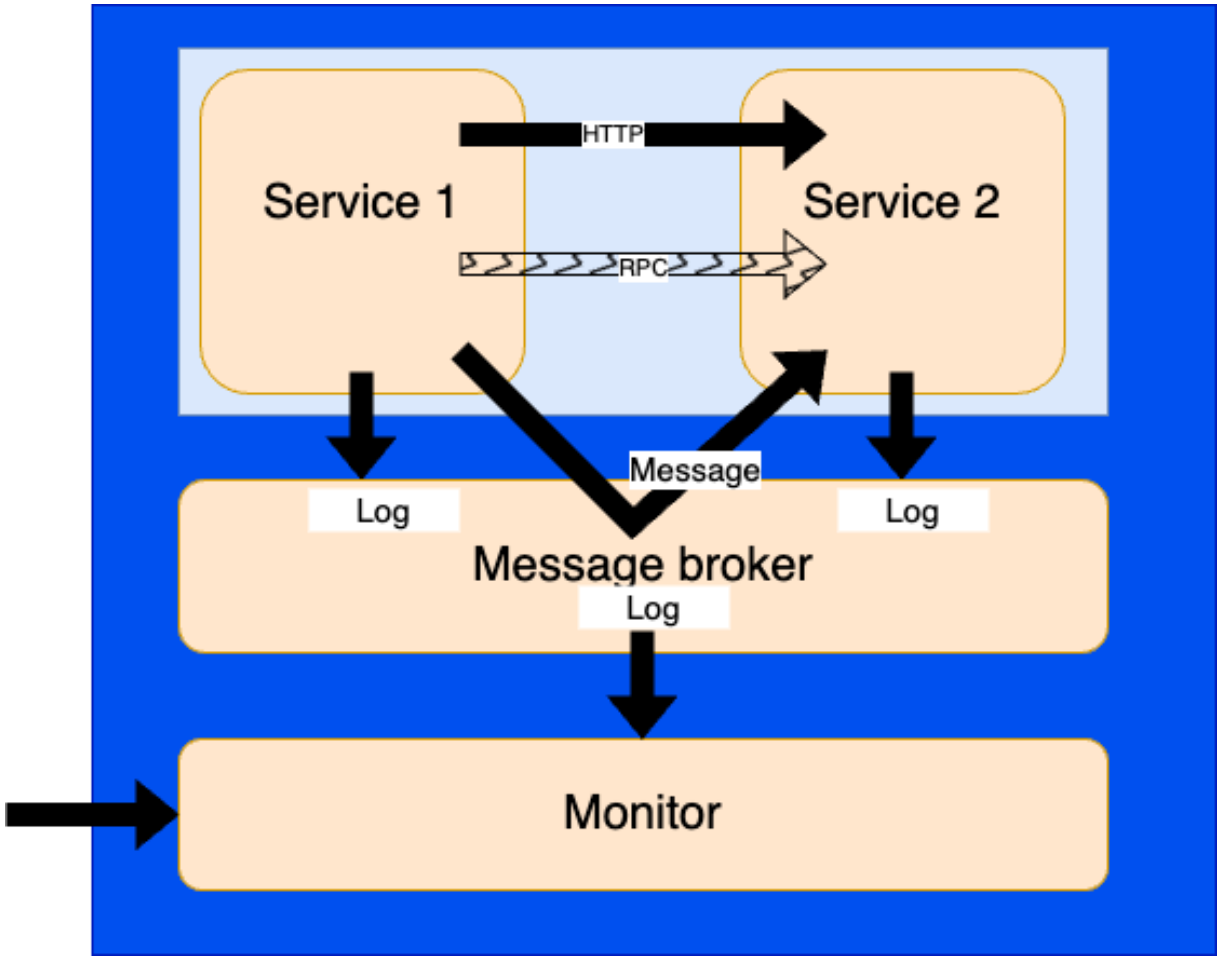
Function of Service 2 (repeated 20 times)



- Wait for 2 seconds
- Establish an HTTP server that listens in port 8000
- Listen topic “message” of the message broker.
- *Optional: establish an RPC (gRPC) service “count0”.*
- *Optional: when the RCP call arrives, the service counts the number of zero-characters (0) in the received string and returns the result as a number.*
- As a response to incoming HTTP message create a new text that adds the remote address (address of service 1). An example:
  SND 1 2022-10-01T06:35:01.373Z 192.168.2.22:8000 192.168.2.21:78390
- Send the above text to message broker (RabbitMQ - topic “log”)
- As a response to incoming message through message broker, create a new text that adds the “MSG” to the string. An example:
  SND 1 2022-10-01T06:35:01.373Z 192.168.2.22:8000 MSG
- Send the above text to message broker (topic “log”)
- Do not stop, but wait for the operator to issue command “docker compose down”.

Tampere University

Function of monitor



- Monitor listens topic “log” of the message broker and keeps the received messages in the memory.
- Monitor also listens GET requests in port 8087, and
- as a response returns the list of received strings from the message broker – MIME-type “text/plain” and each message on a separate line.

Notes

- As the service should run in separate containers, you should write *Dockerfiles* for the all services and *docker-compose.yaml* to start both containers and connect then with a private network.
- In docker-compose.yaml make the port 8087 visible to outside, too. (but not other ports)
- For fluent testing by teaching staff you should do your best to ensure that that both docker images and running containers have unique names from your other students.
- Start a new GIT branch “exercise2” – do not change files in exercise1!

Tampere University

Submission

Tampere University

Grading

The points from this exercise depend on timing and content:

- Maximum 8 standard points are given.
- Missing the first deadline (23.10.2023): points reduced by 1 points / starting day. The absolute deadline is 30.10.2023.
- How well the requirements (including technical instructions to the submit your project) are met: 6p
- Following the good programming and docker practices: 2p