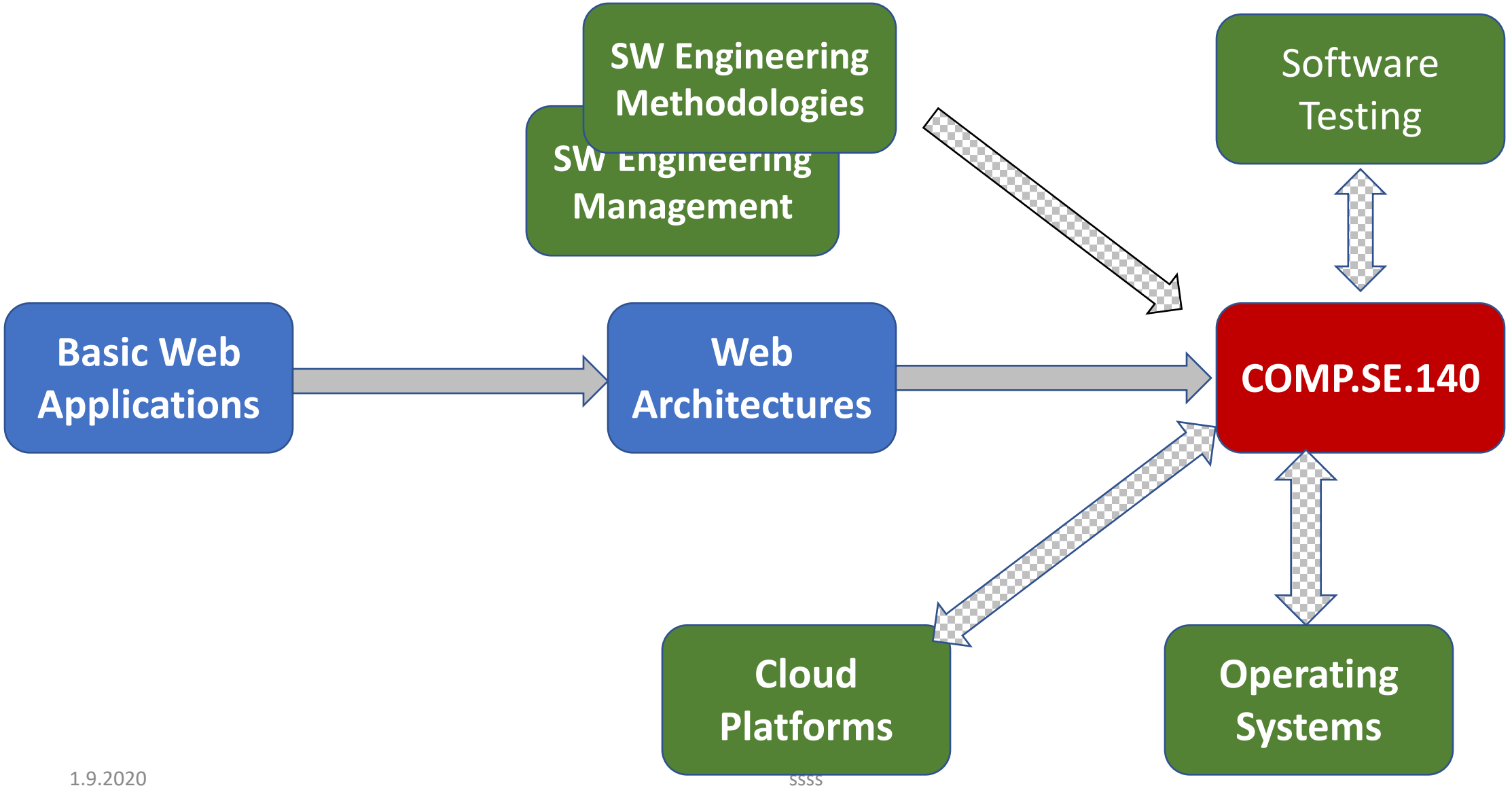
Tampereen yliopisto
Tampere University

Lecture 2
virtualization

Tampereen yliopisto
Tampere University

Content

- General course related matters
- Recap of DevOps
- Virtualization – what and why?
  - Software engineer point of view
- Containers
- Containers in practice: Docker
- Summary


SW Engineering Methodologies
SW Engineering Management
Software Testing
Basic Web Applications
Web Architectures
COMP.SE.140
Cloud Platforms
Operating Systems

Tampereen yliopisto
Tampere University

General course related matters

Tampereen yliopisto
Tampere University

Numbers & statistics

- Combine list from Sisu and Plus: 89
- Sign-ups in plus: 85
- Responses to initial questionary: 36 (still time left)
  - Heterogenous background
  - Some students lack relevant courses
  - Most responders work full- or part-time
  - Surprisingly, many students have taken Cloud Platforms

Tampereen yliopisto
Tampere University

Communication

- Fast messages: email
  - If you have not received, let me know
- Static info: plussa
- Sisu is updated seldom, if ever.

Tampereen yliopisto
Tampere University

DevOps practices

- Organizational
  - increased scope of responsibilities for developers;
  - intensified cooperation between development and operations.

- Technical
  - automation,
  - monitoring
  - measurement

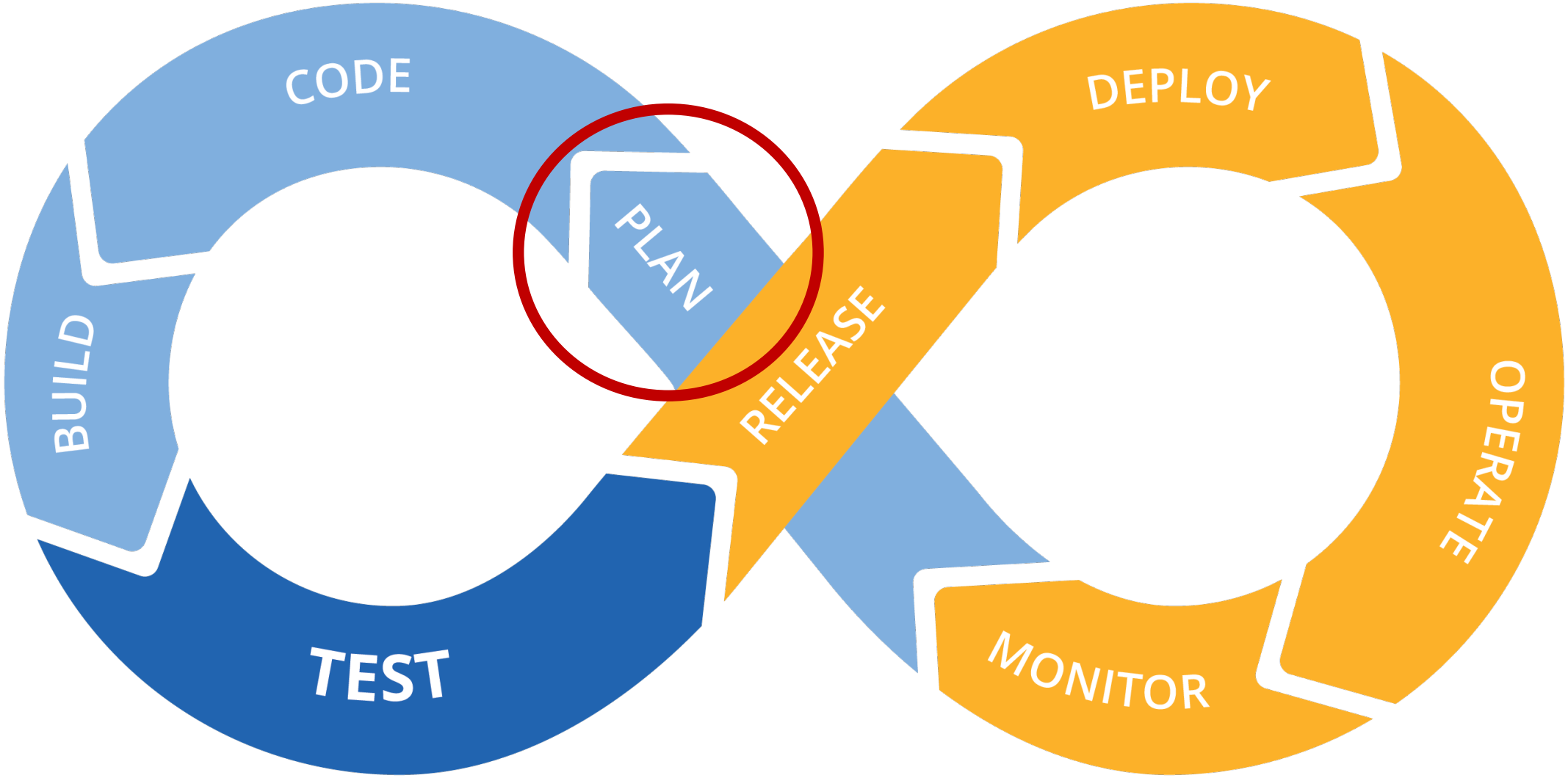
Tampereen yliopisto
Tampere University

What is DevOps (there are several definitions)

- Lucy Lwakatare:
  - DevOps is a concept that embodies a **cultural and mindset change** that is substantiated with a **set of practices** to encourage **cross-disciplinary collaboration between software development and IT operations** within a software company. The main purpose for the collaboration is to enable the **fast release of quality software changes** while simultaneously **operating resilient systems**.
  - From a **socio-technical perspective**, DevOps practices are focused on the **automation practices** of software deployment and infrastructure management, specifically automation of configuration management and monitoring.

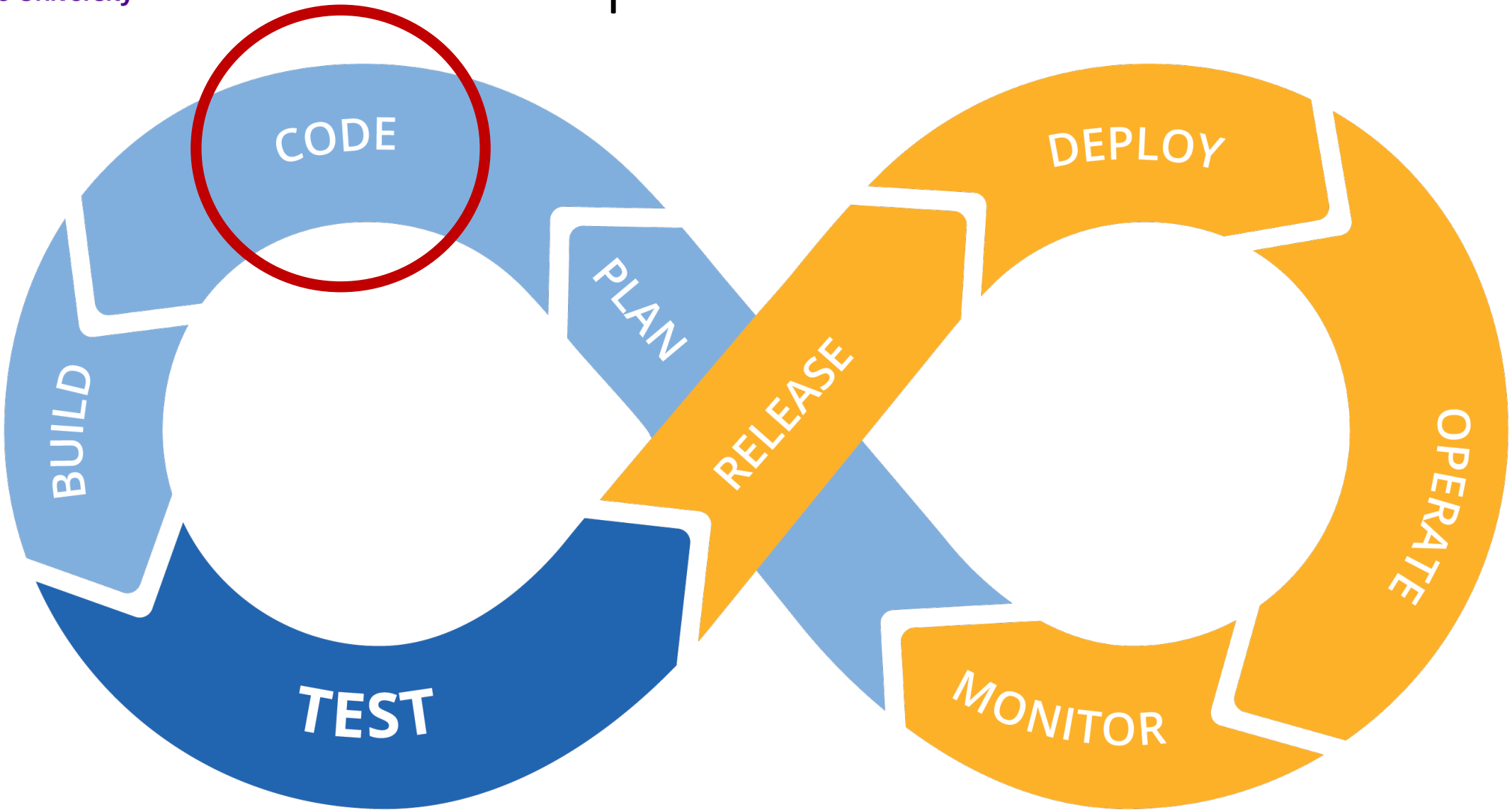
Tampereen yliopisto
Tampere University

DevOps



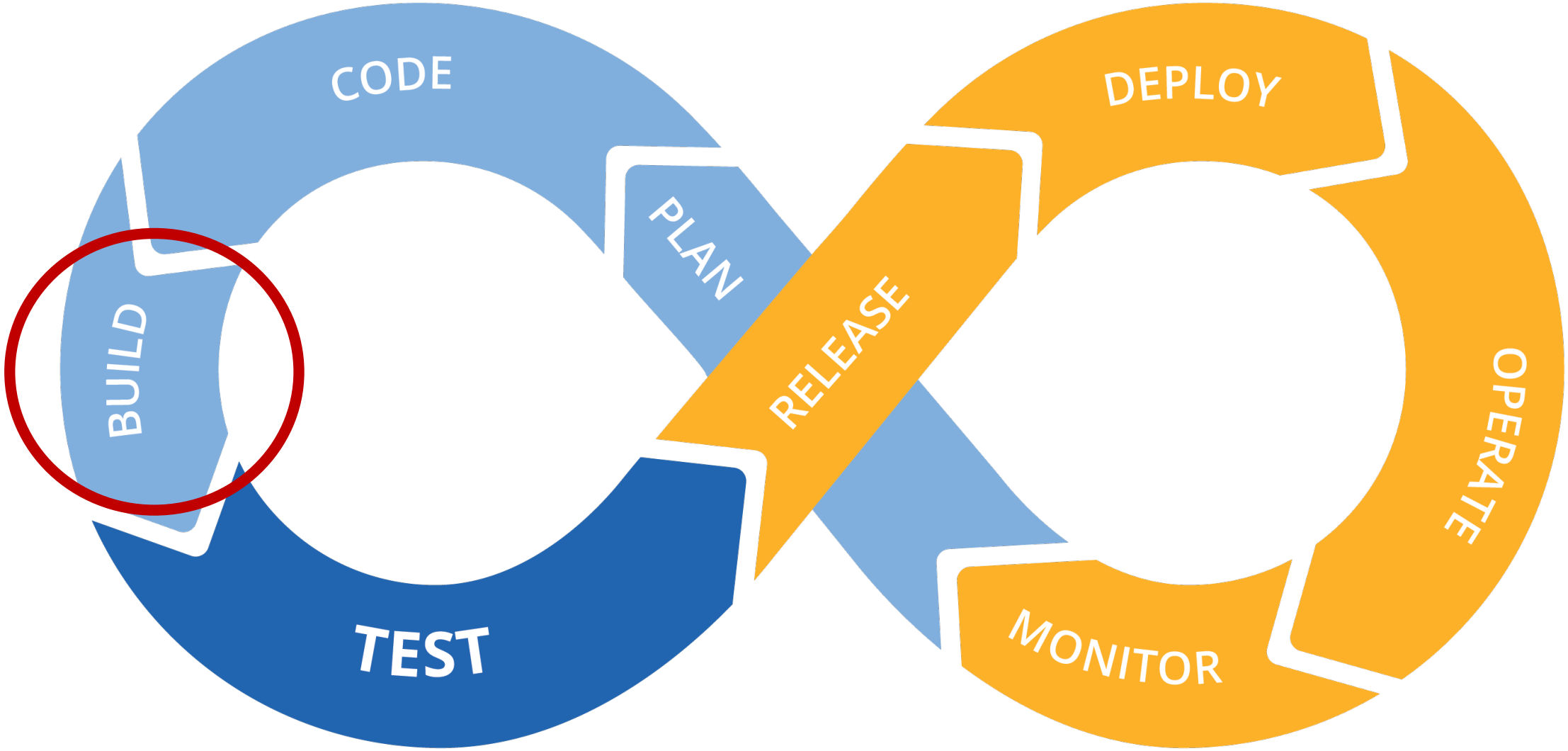
Tampereen yliopisto
Tampere University

DevOps



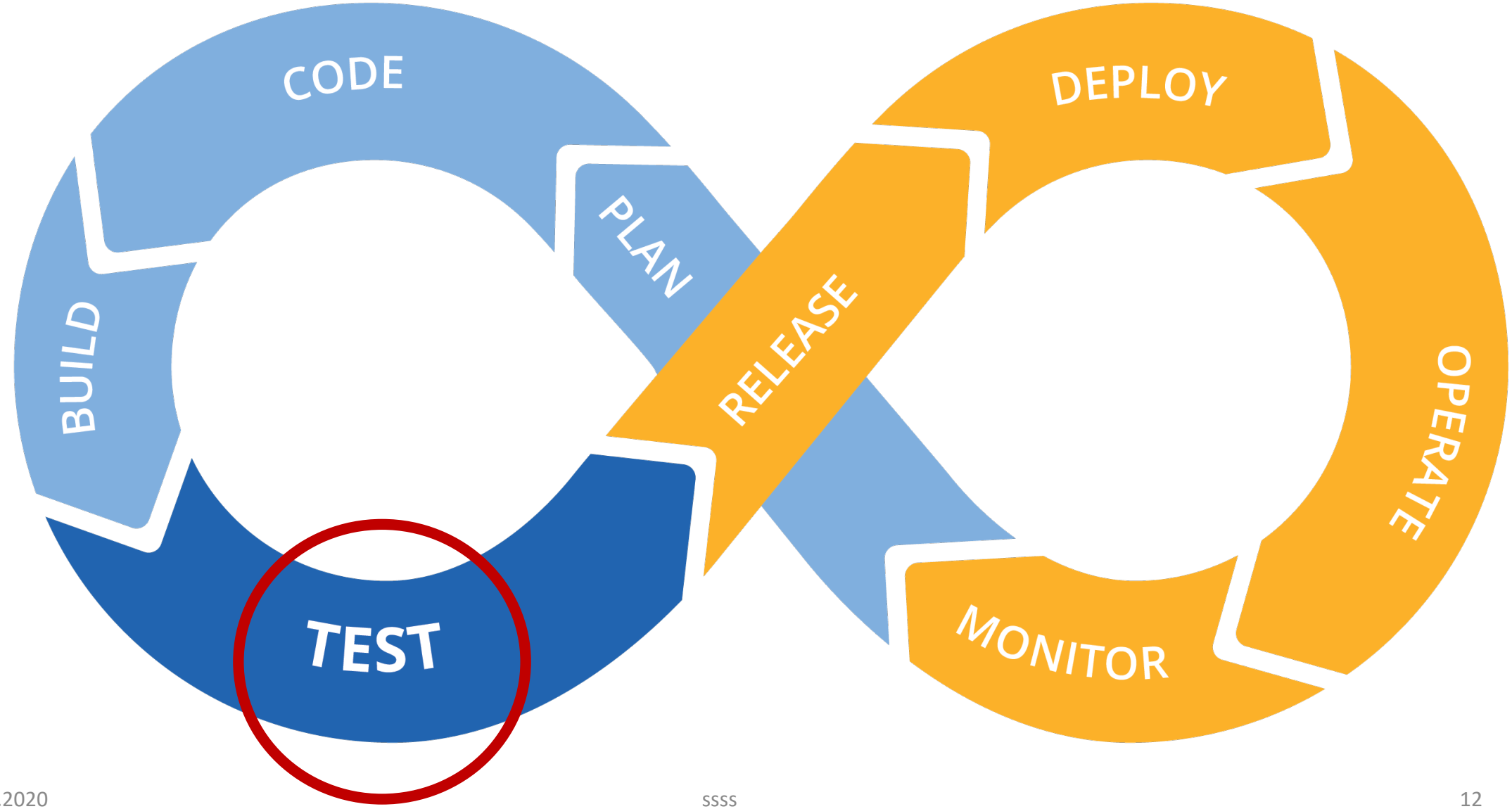
Tampereen yliopisto
Tampere University

DevOps



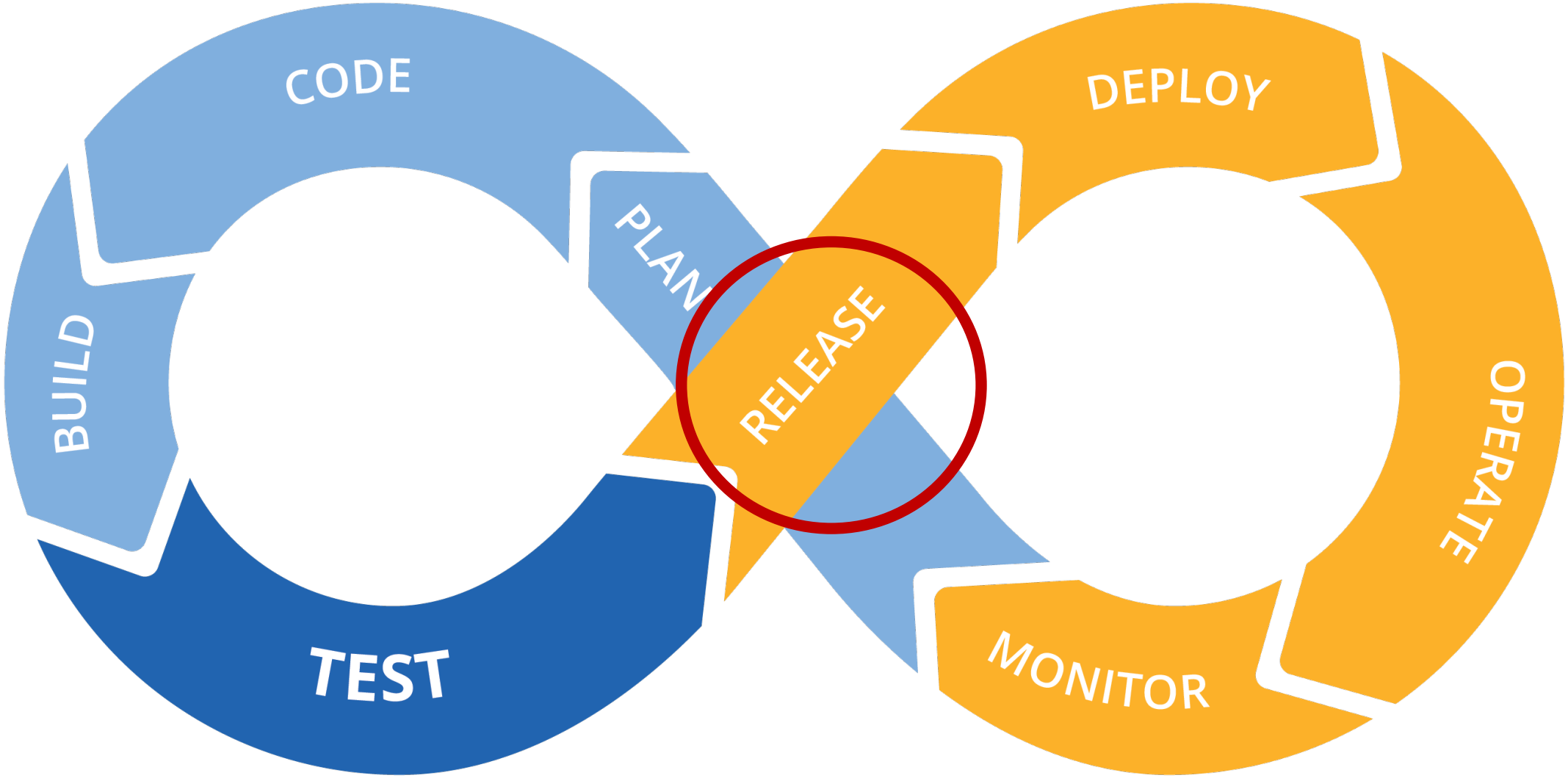
Tampereen yliopisto
Tampere University

DevOps



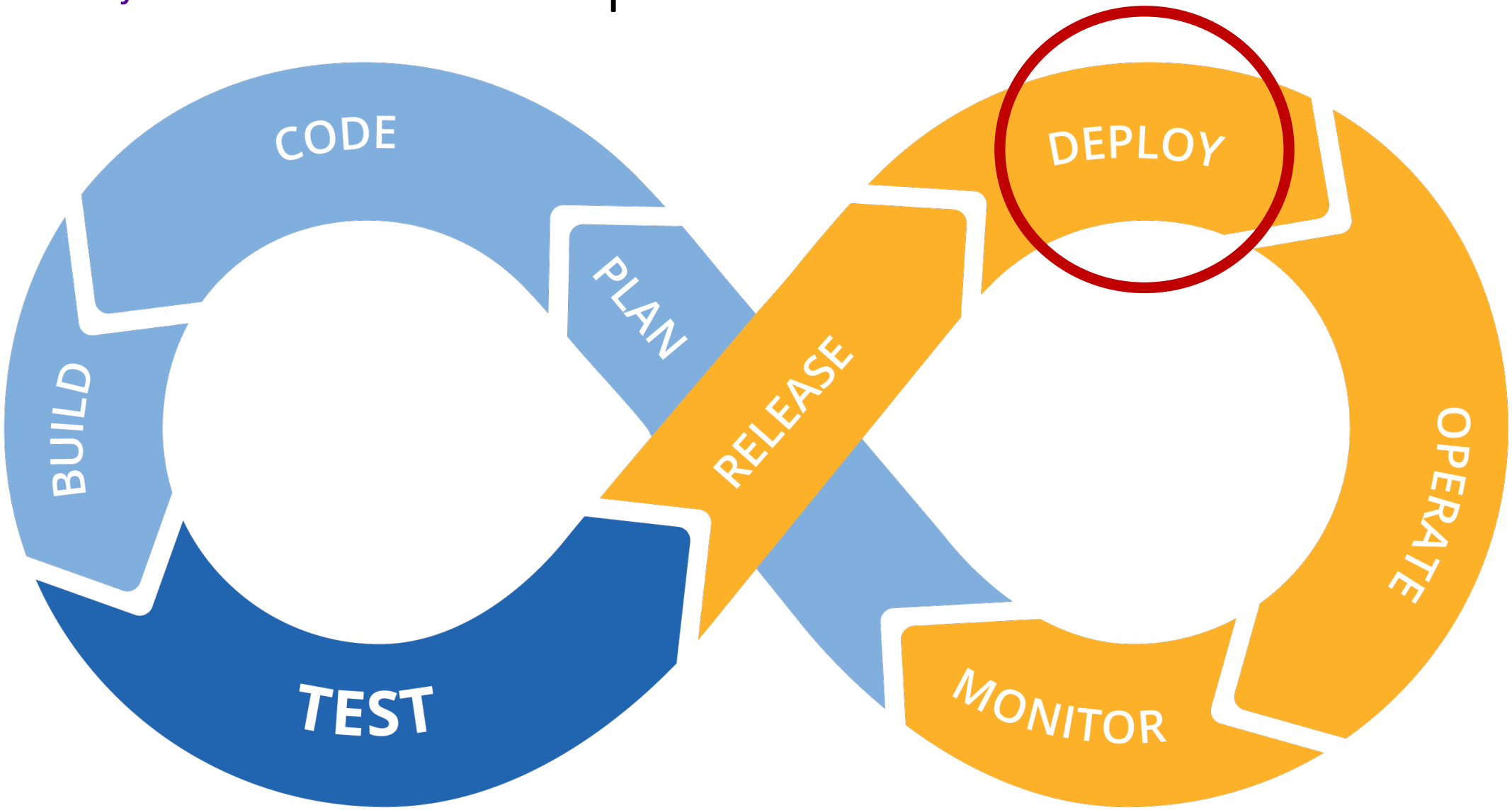
Tampereen yliopisto
Tampere University

DevOps



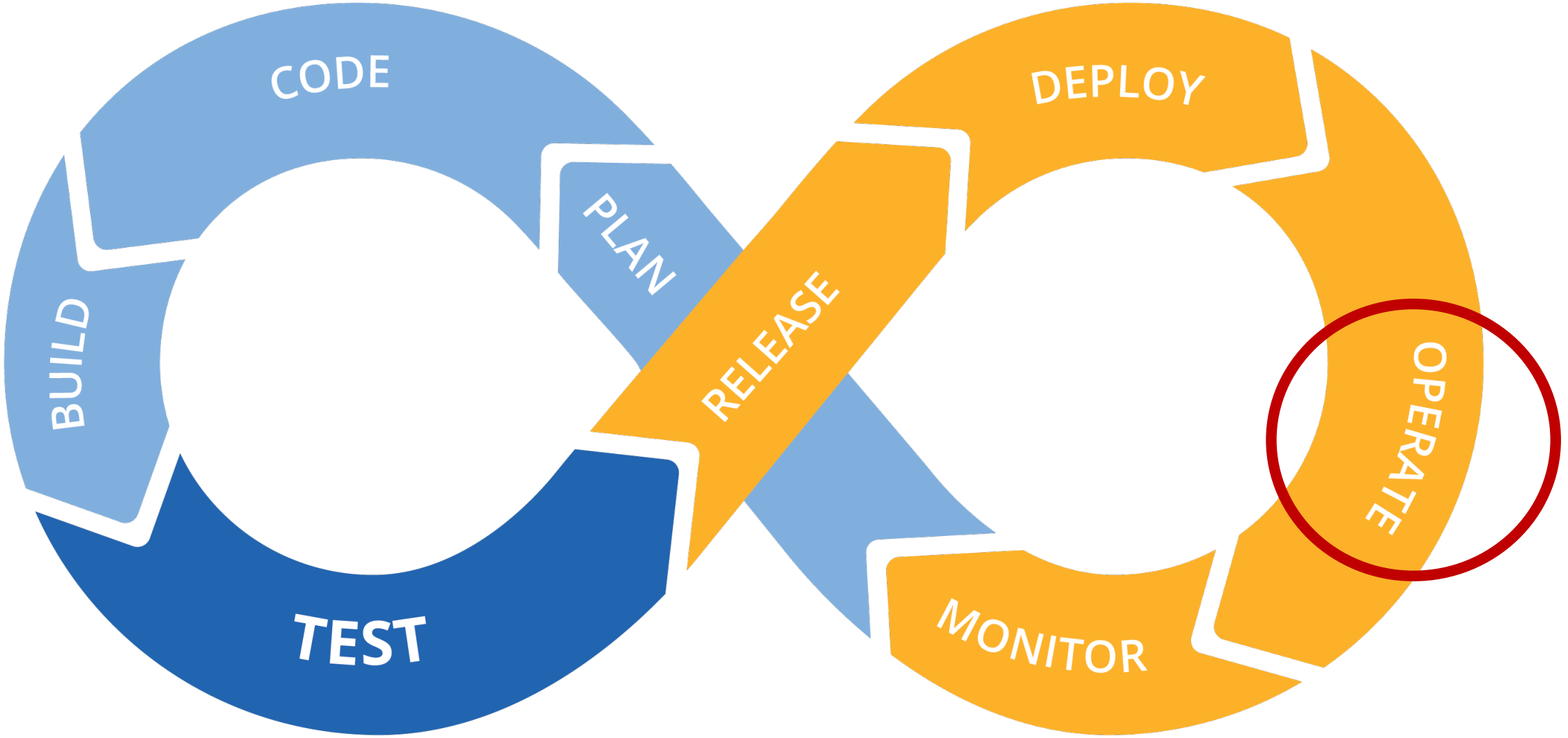
Tampereen yliopisto
Tampere University

DevOps



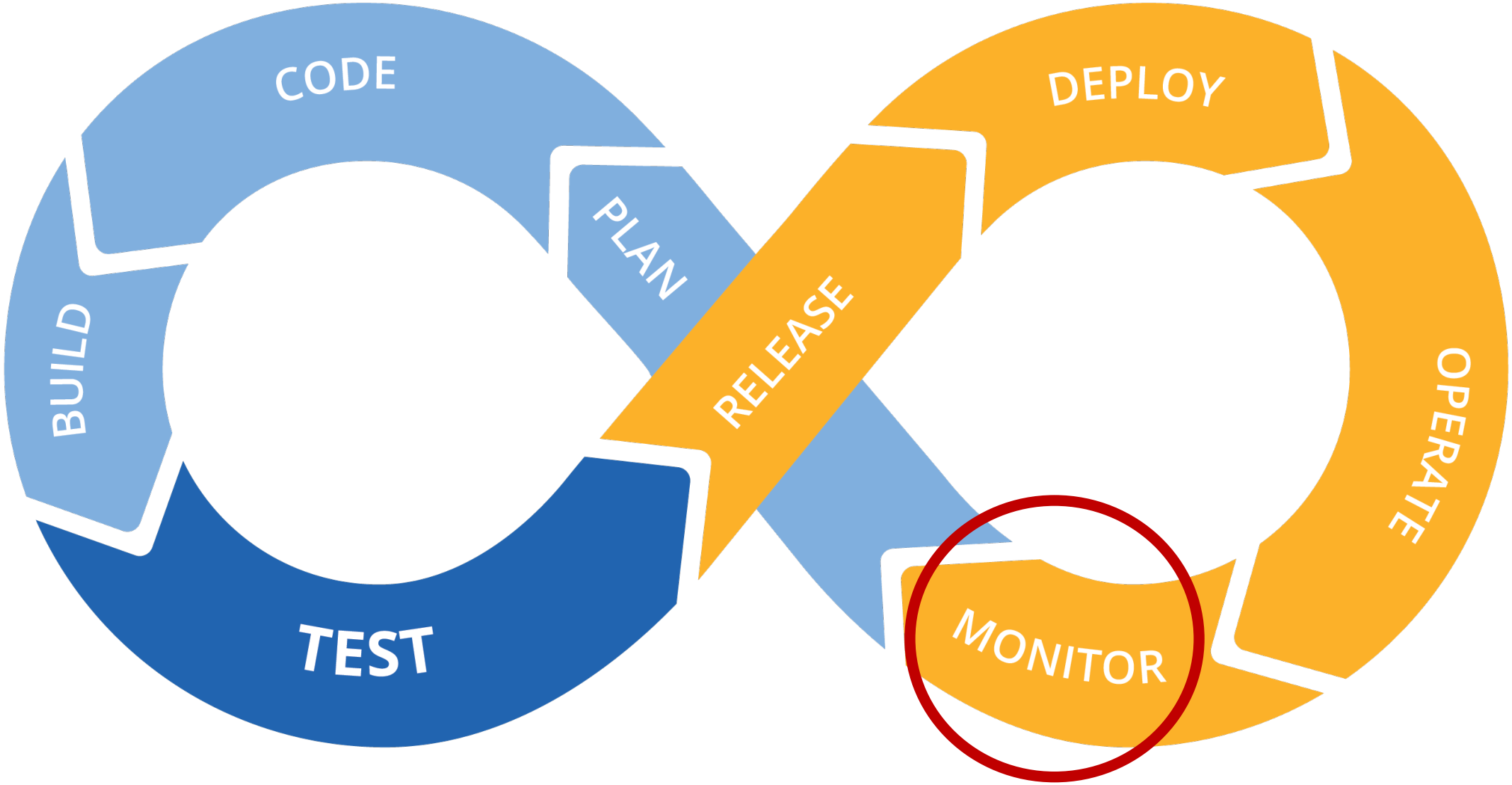
Tampereen yliopisto
Tampere University

DevOps

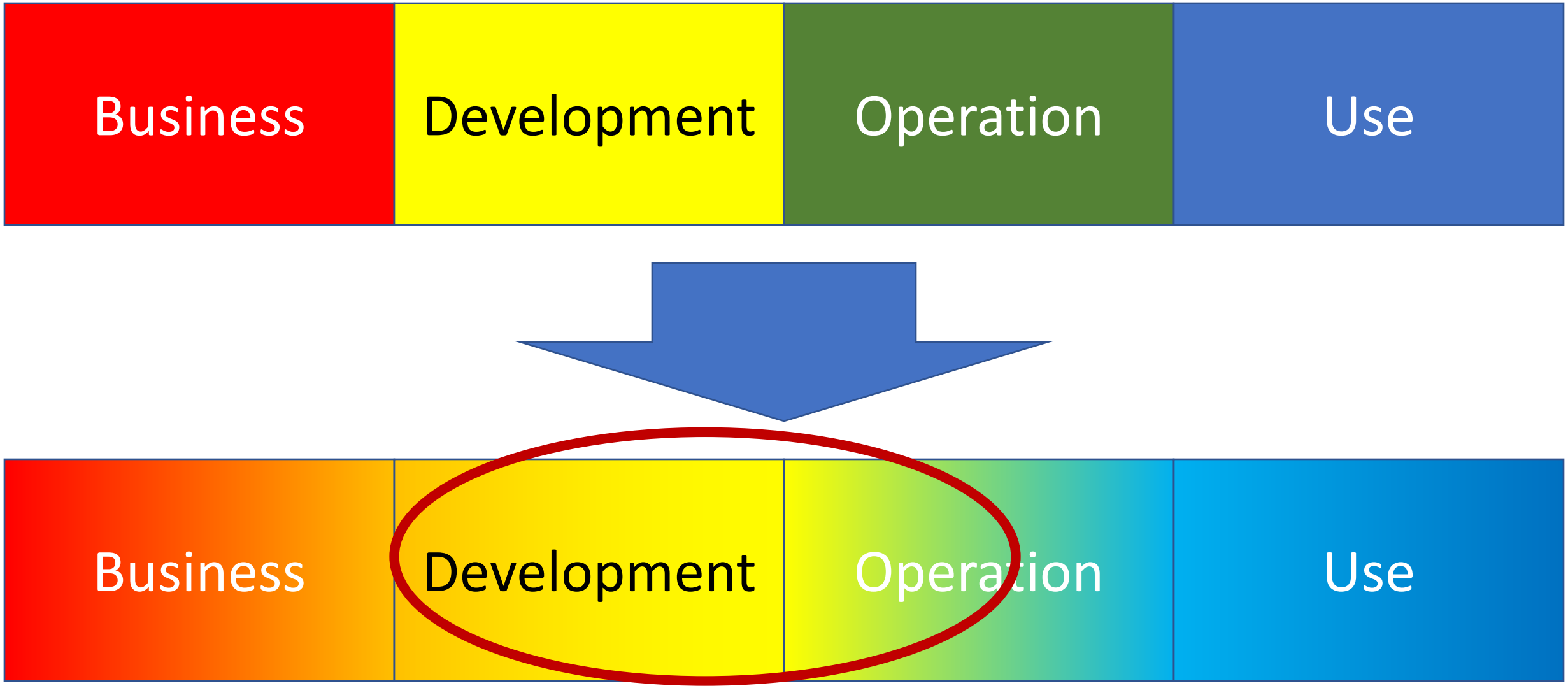


Tampereen yliopisto
Tampere University

DevOps



Tampereen yliopisto
Tampere University

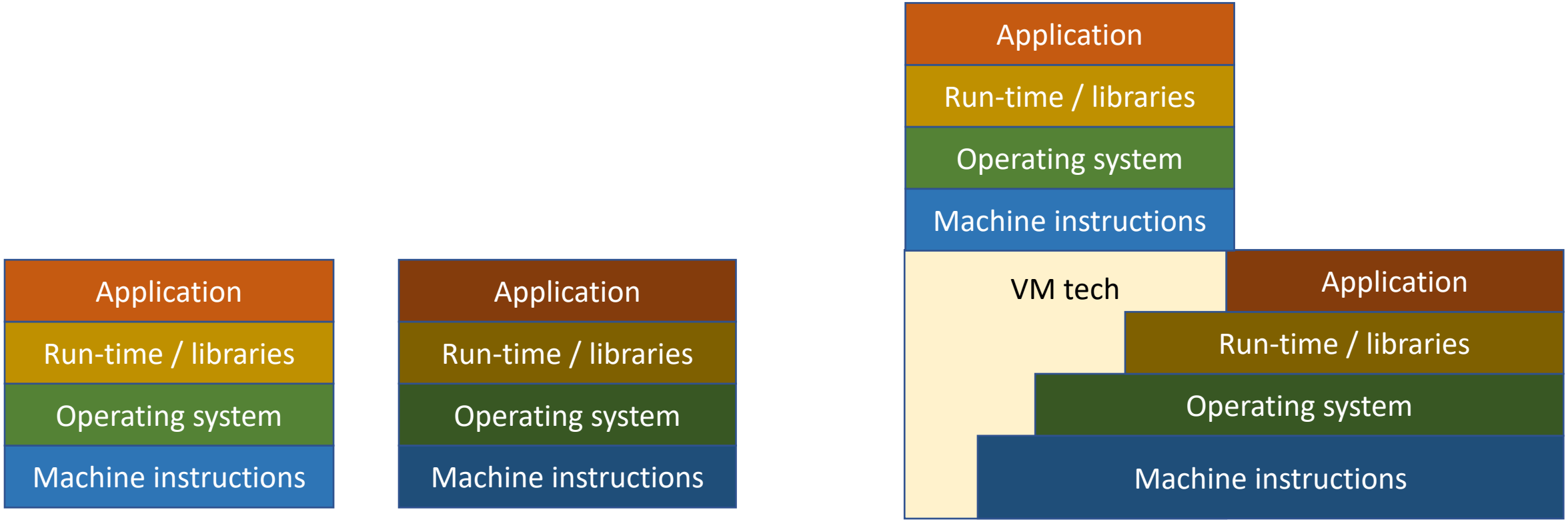


Tampereen yliopisto
Tampere University

Virtualization – what and why?

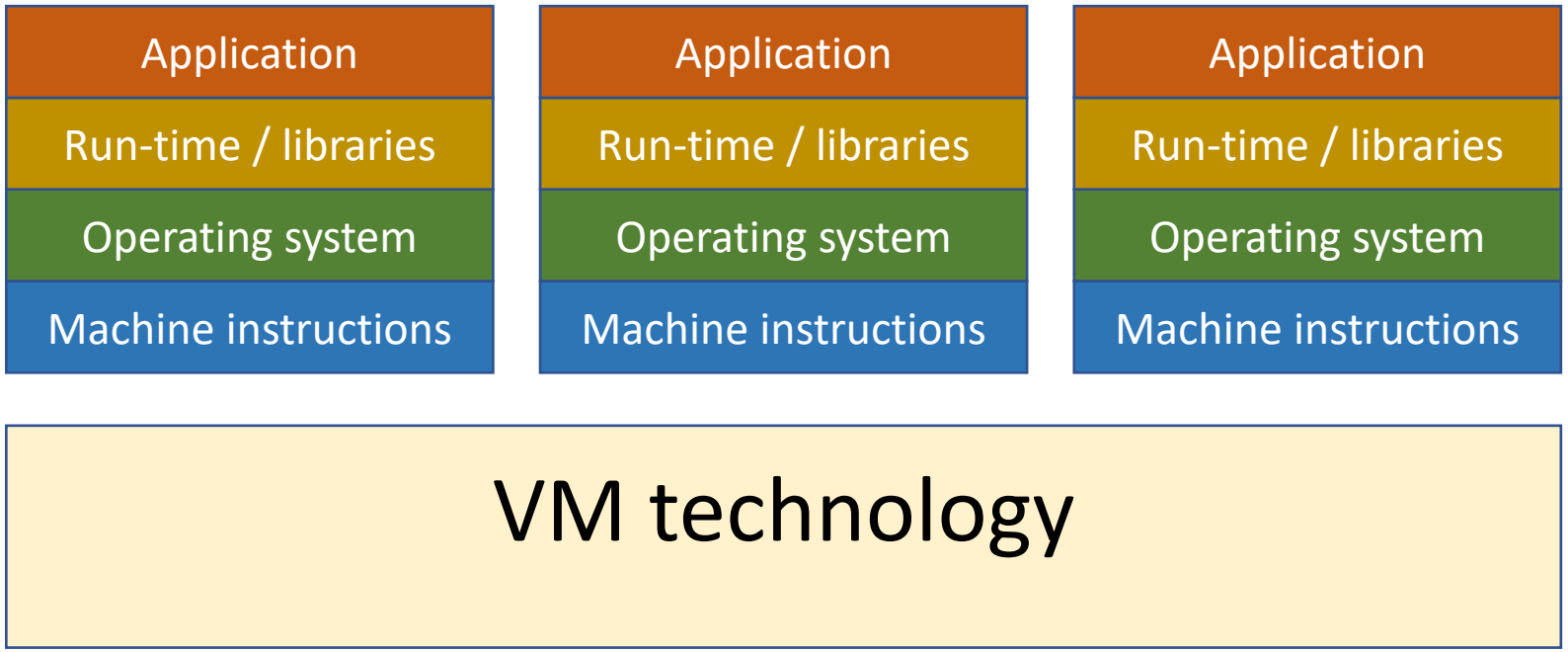
Tampereen yliopisto
Tampere University

Use case 1: run ”foreign” software

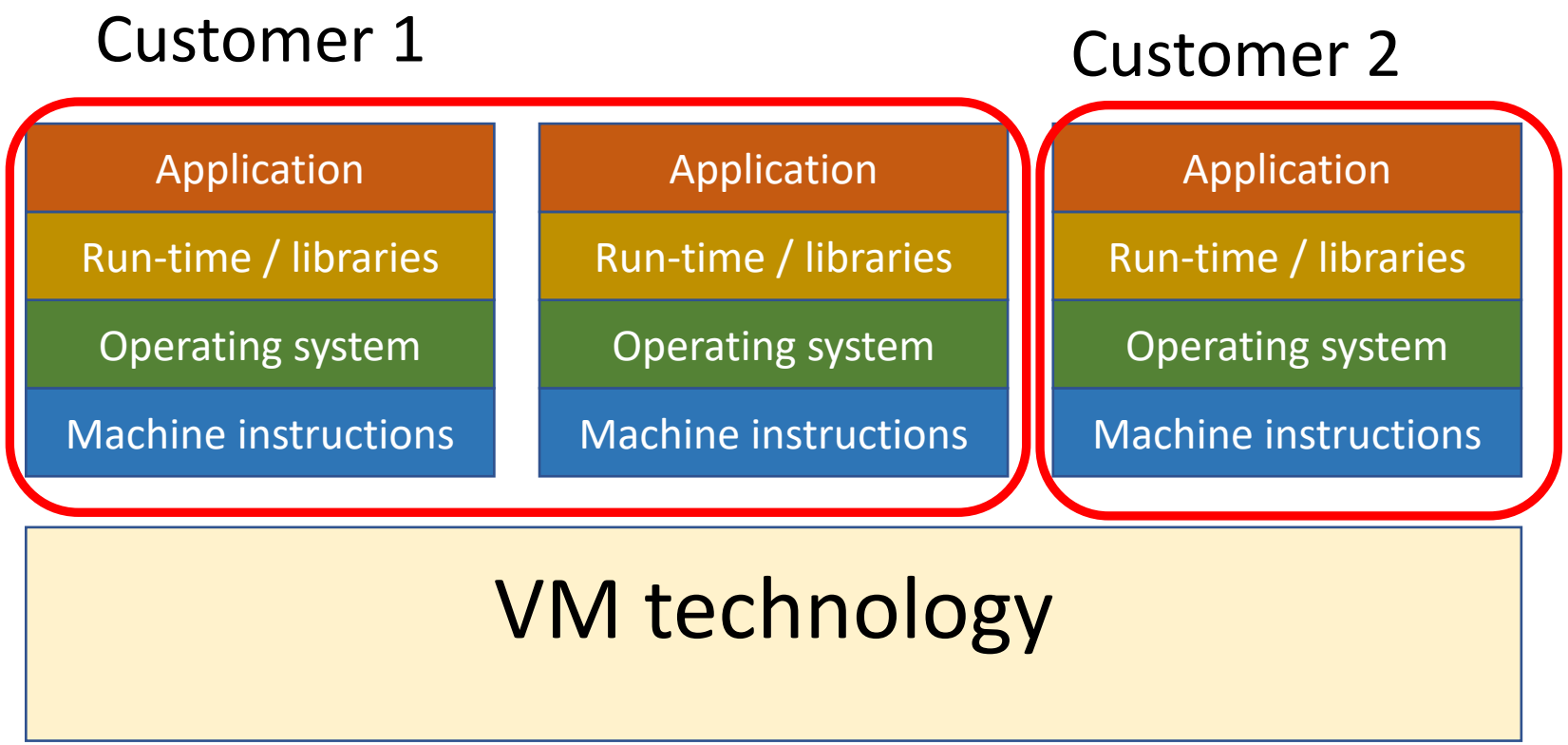


Tampereen yliopisto
Tampere University

Use case 2: isolate



Use case 3: scale



Tampereen yliopisto
Tampere University

Levels of virtualization

- Hardware virtualization
- Operating system virtualization
- Desktop virtualization
- Application virtualization
- Network virtualization

Tampereen yliopisto
Tampere University

Network virtualization

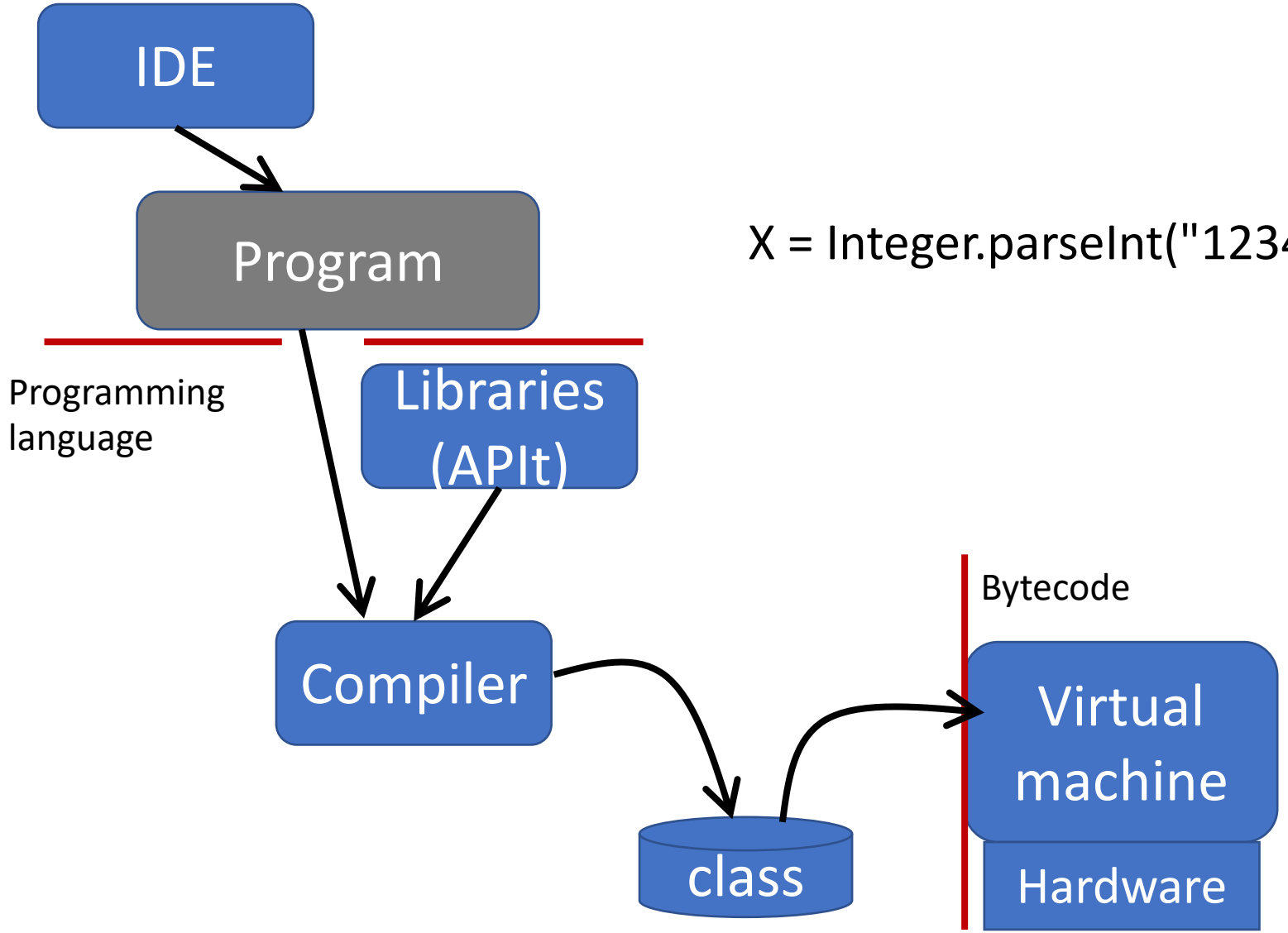
- Network, its HW and protocols, is simulated with software so that it looks like a different network to applications
- Different from OSI layer models

- Is VPN a virtual network?

Application virtualization

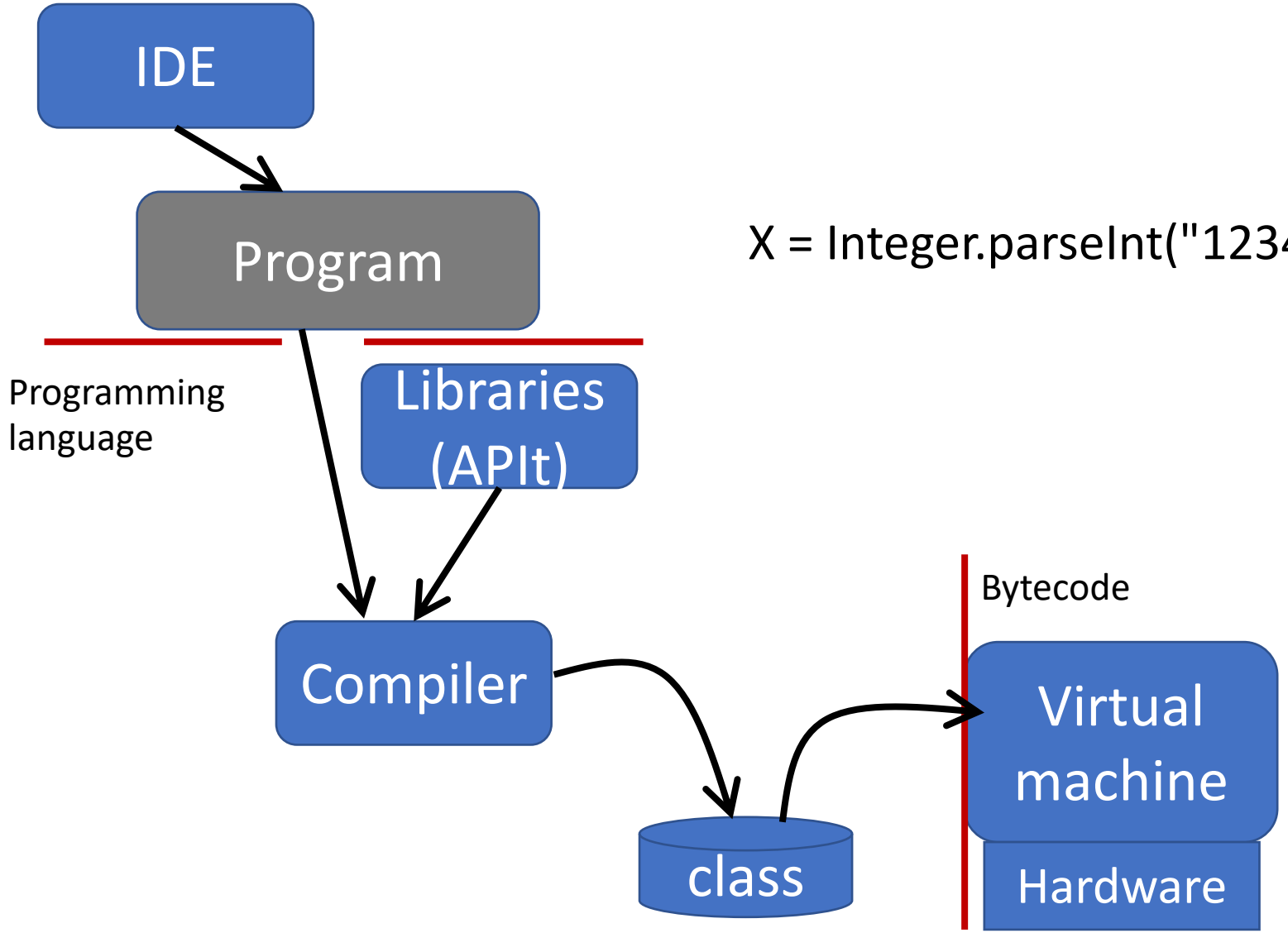
- Applications are compiled to machine-independent ”machine ” code
- Applications are run with a virtual machine
- Benefits
  - Same code can be run on different CPUs
  - Increased safety. Why?
- Problems
  - Performance

Example: java




```
X = Integer.parseInt("1234");
```

Java security




```
X = Integer.parseInt("1234");
```

```
$ javap -c test
Compiled from "test.java"
class test {
  int X;


  test();
    Code:
       0: aload_0             // this
       1: invokespecial #1    // Method java/lang/Object."<init>":()V
       4: return


  void foo();
    Code:
       0: aload_0
       1: ldc           #2    // String 1234
       3: invokestatic  #3    // Method java/lang/Integer.parseInt:(Ljava/lang/String;)I
       6: putfield      #4    // Field X:I
       9: return
}
```

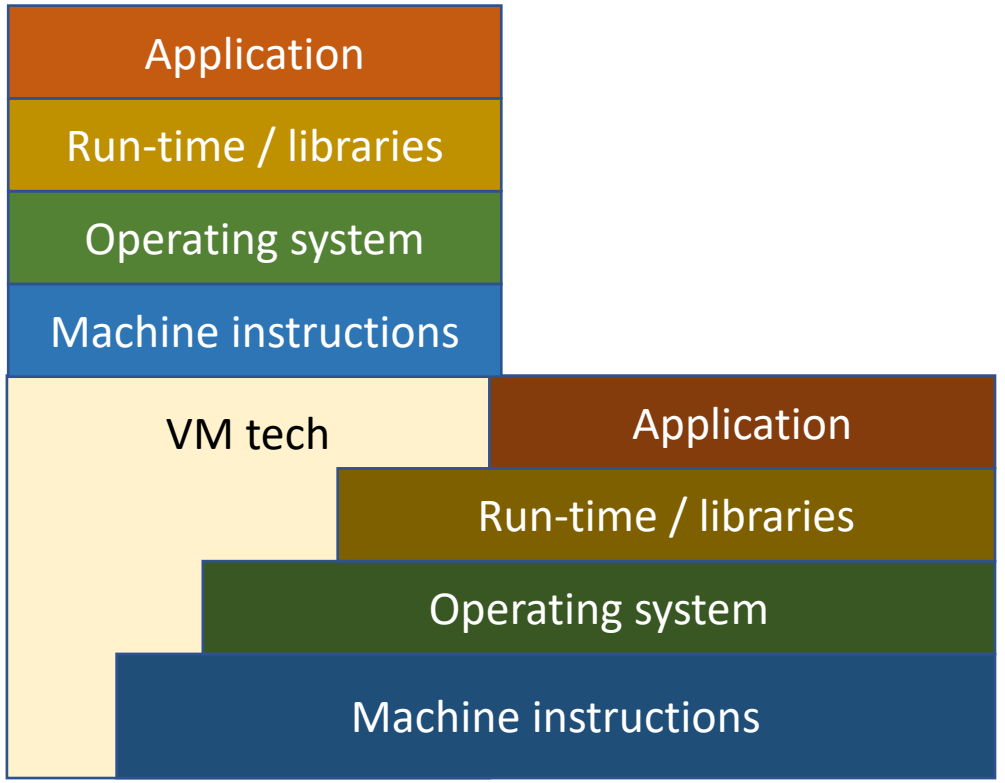
Tampereen yliopisto
Tampere University

Desktop virtualization

- When you run the ”linux-desktop” (linux-desktop.cc.tut.fi) on your windows machine

Hardware virtualization

- Hypervisor
  - HW and/or SW based

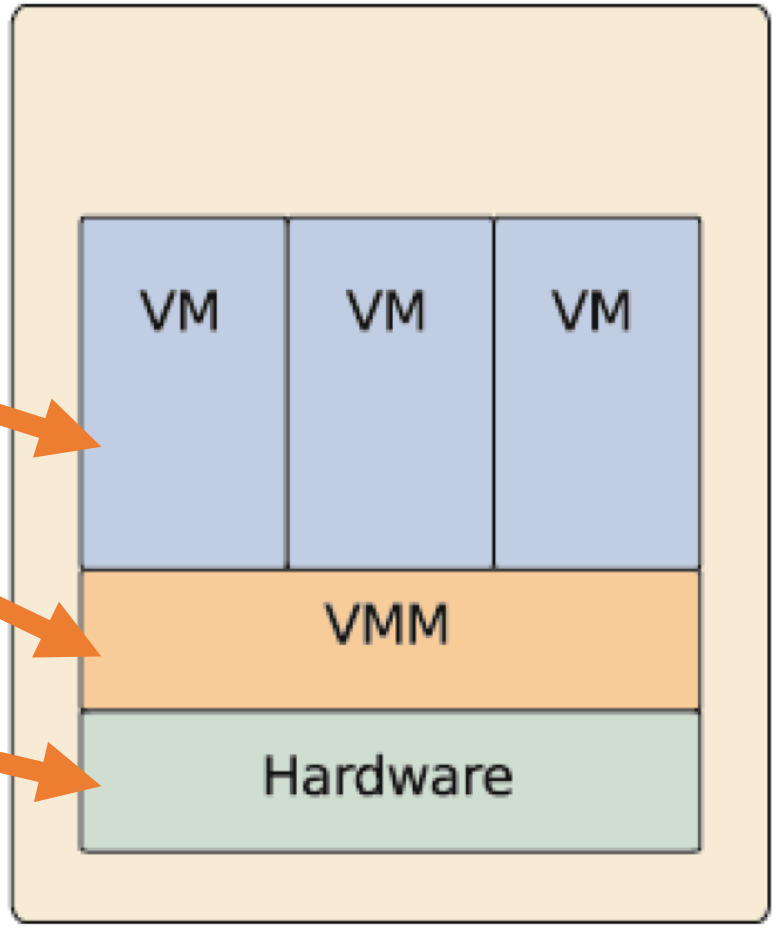


Tampereen yliopisto
Tampere University

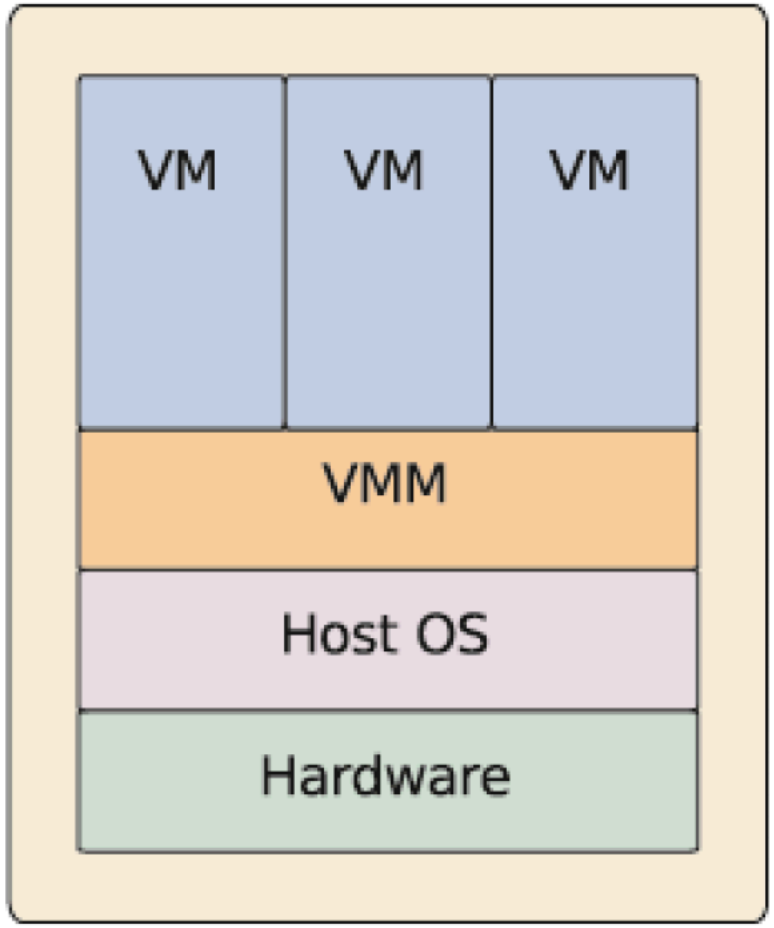
Guest

Virtual Machine Mngr
(Hypervisor)

Host



Hypervisor Type I



Hypervisor Type II

Mika Kaaretkoski,
Master thesis, 2018

F. Rodrguez-Haro et al., “A summary of virtualization techniques,” Procedia Technology, vol. 3, pp. 267 { 272, 2012, the 2012 Iberoamerican Conference on Electronics Engineering and Computer Science. [Online]. Accessed: 4.8.2018 Available: http://www:sciencedirect:com/science/article/pii/S2212017312002587

Type-1, native or bare-metal hypervisors

- Examples of virtualization that uses hardware assisted are Kernel-based Virtual Machine (KVM), VirtualBox, Xen, Hyper-V, and VMware products
- Hardware-assisted
- Pros & cons
  + efficient
  + provides service to all the guests in equivalent way
  - requires cleaning of the existing system clean
  - I/O device drivers must be available for installation in the VMM

Different approached (source of pictures: VMWARE)

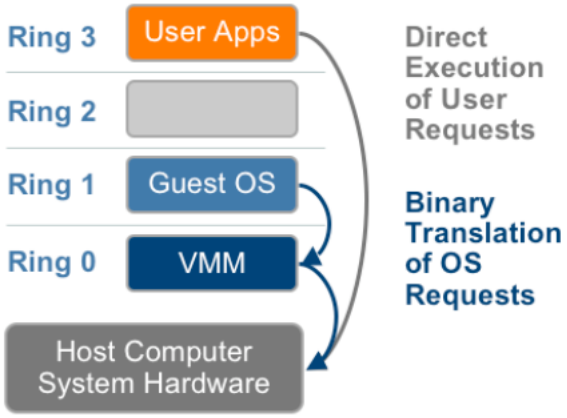
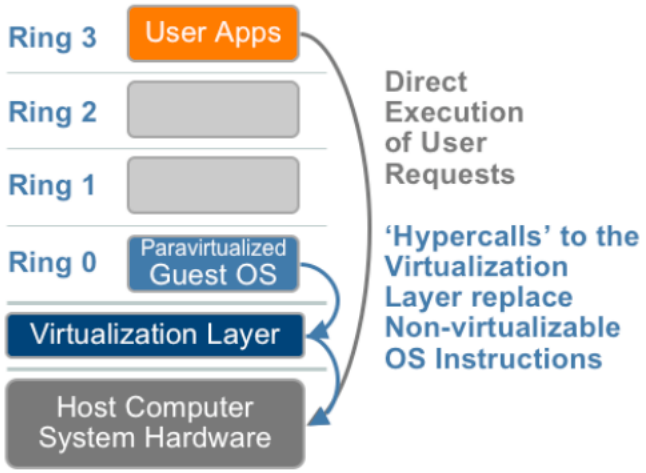


Figure 5 – The binary translation approach to x86 virtualization



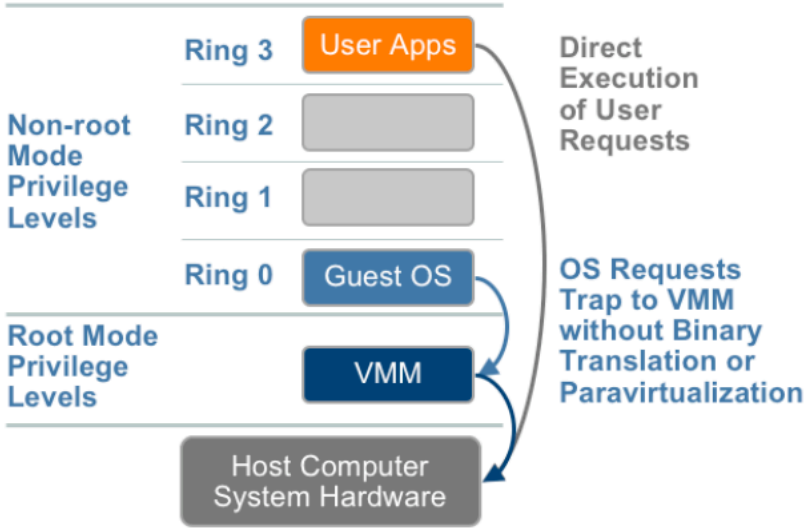


Figure 7 – The hardware assist approach to x86 virtualization

Skipped 2020

Type-2 or hosted hypervisors

- A guest operating system runs as a process on the host.
- VMware Workstation, VMware Player, VirtualBox, Parallels Desktop for Mac and QEMU
- Emulation or binary translation
- Pros & cons
  + Virtualizing SW use device drivers and other lower-level services of host
  - Loss of efficiency because more layers of SW involved

Tampereen yliopisto
Tampere University

Containers

Tampereen yliopisto
Tampere University

Basic of containers

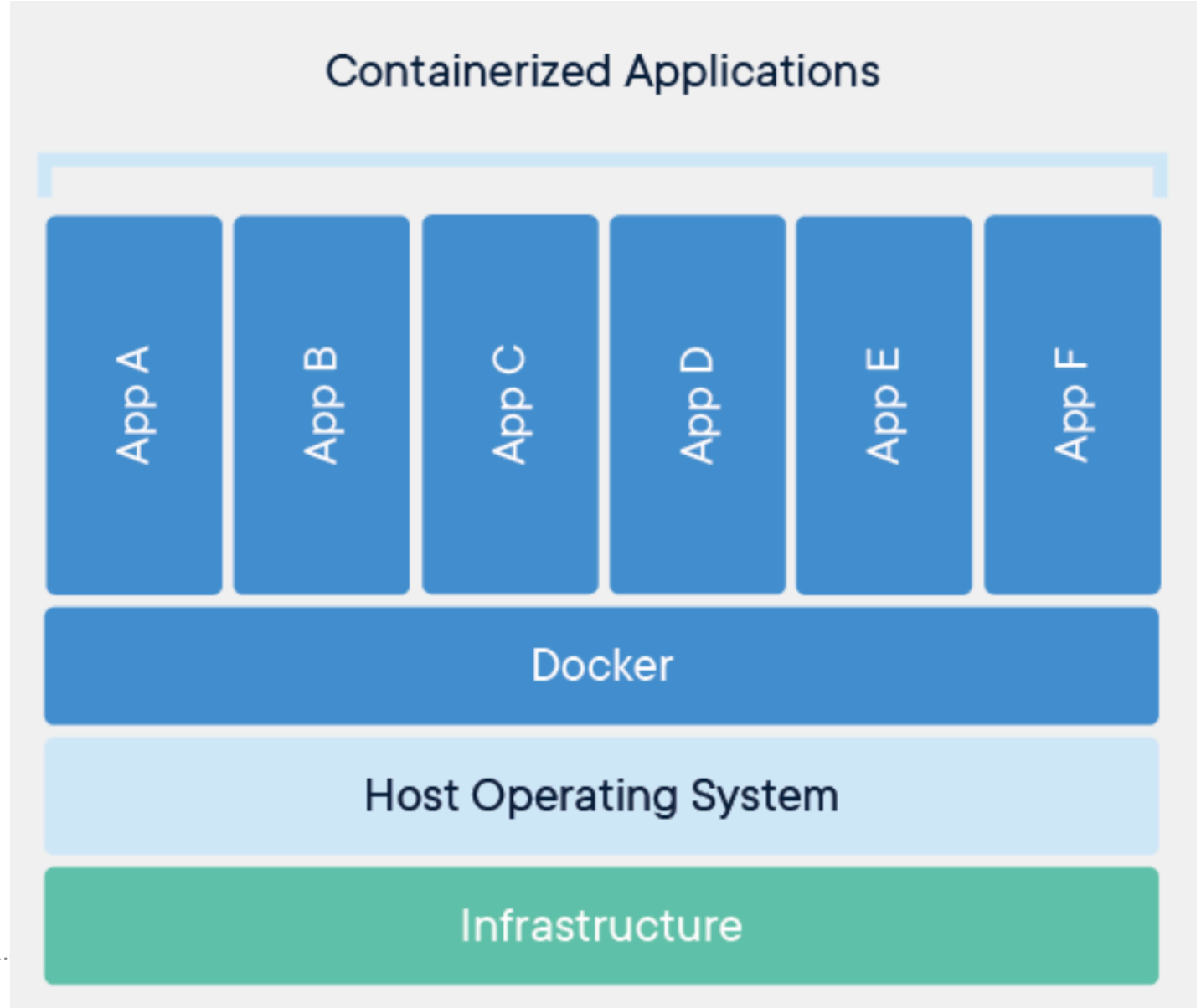
- Lightweight virtualization
- Guests share OS kernel with host
- In Linux (LXC) basically Separate namespaces
  - IPC – inter-process communication
  - Network
  - Mount – Filesystem
  - PID – Processes
  - User
  - UTS – hostname and domains
  - Cgroup –
- Nice tutorial (with commercial at the end) https://www.youtube.com/watch?time_continue=2&v=n-JwAM6XF88

Tampereen yliopisto
Tampere University

The dominant way to manage containers: Docker

So dominant that we go details of one technology

From docker.com



Tampereen yliopisto
Tampere University

Use case example

- Your application needs
  - Certain version of nodejs
  - Set of libraries (certain versions)
  - Mongo database

- Your system has
  - Wrong version of nodejs
  - Mongo serving another application

- Solution
  - Create a docker image (container)
  - Install the image
  - Run the image

Tampereen yliopisto
Tampere University

https://www.katacoda.com/courses/container-runtimes

(thanks to your fellow student 2019)

```
Terminal +
Your Interactive Bash Terminal. A safe place to learn and execute commands.


$
$ docker pull redis:3.2.11-alpine
3.2.11-alpine: Pulling from library/redis
ff3a5c916c92: Pull complete
aae70a2e6027: Pull complete
87c655da471c: Pull complete
bc3141806bdc: Pull complete
53616fb426d9: Pull complete
9791c5883c6a: Pull complete
Digest: sha256:ebf1948b84dcaaa0f8a2849cce6f2548edb8862e2829e3e7d9e4cd5a324fb3b7
Status: Downloaded newer image for redis:3.2.11-alpine
$
```

Tampereen yliopisto
Tampere University

Let investigate a bit

```
$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
redis               3.2.11-alpine       ca0b6709748d        15 months ago       20.7MB
```

Tampereen yliopisto
Tampere University

I could not resist

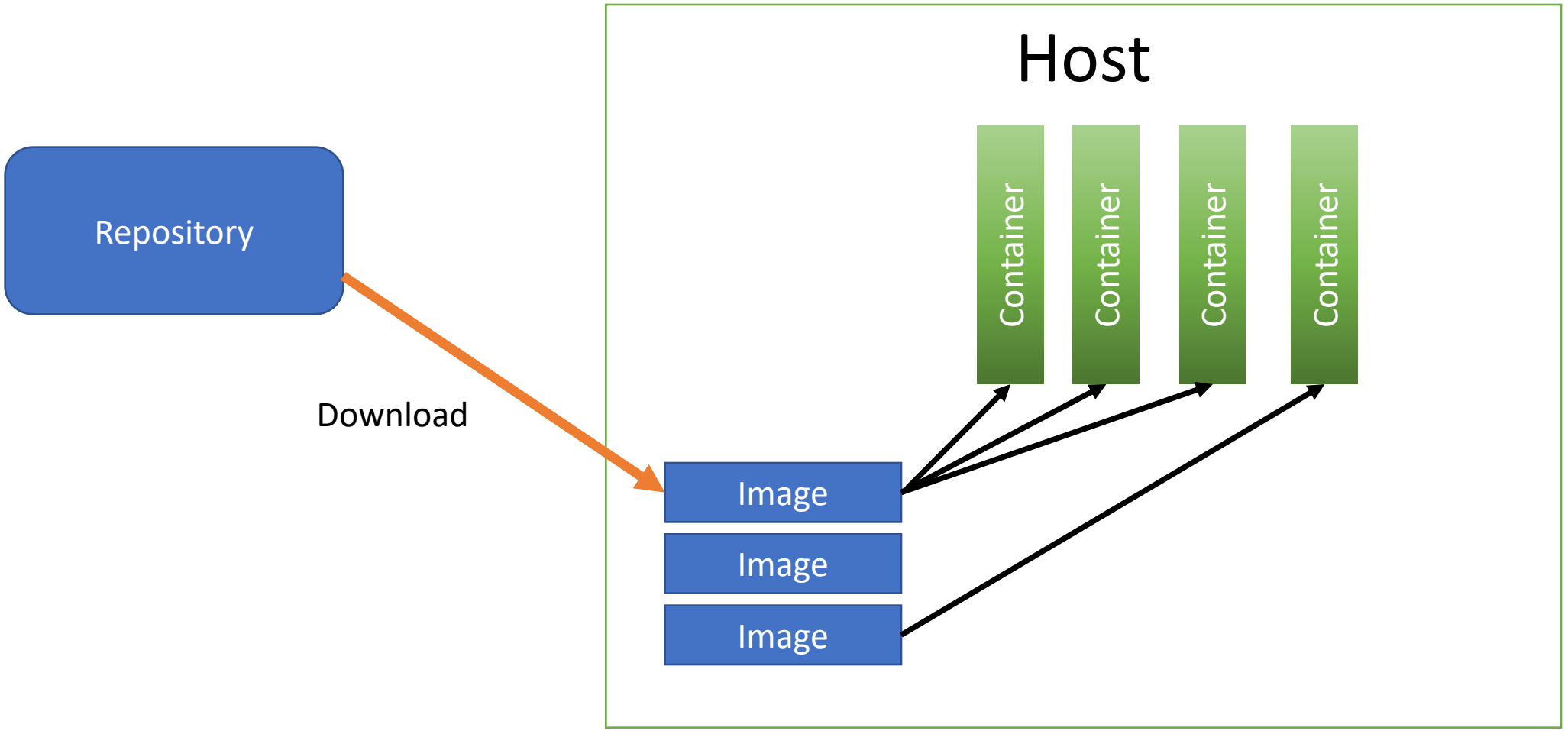
```
$ docker run ca0b6709748d
1:C 05 Sep 17:27:18.972 # Warning: no config file specified, using the default config. In order to s
s-server /path/to/redis.conf
                _.-
            _.-``__ ''-._
       _.-``    `.  `_.  ''-._           Redis 3.2.11 (00000000/0) 64 bit
   .-`` .-```.  ```\/    _.,_ ''-._
  (    '      ,       .-`  | `,    )     Running in standalone mode
  |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
  |    `-._   `._    /     _.-'    |     PID: 1
   `-._    `-._  `-./  _.-'    _.-'
  |`-._`-._    `-.__.-'    _.-'_.-'|
  |    `-._`-._        _.-'_.-'    |           http://redis.io
   `-._    `-._`-.__.-'_.-'    _.-'
  |`-._`-._    `-.__.-'    _.-'_.-'|
  |    `-._`-._        _.-'_.-'    |
   `-._    `-._`-.__.-'_.-'    _.-'
```

Tampereen yliopisto
Tampere University

What did we just see?

Tampereen yliopisto
Tampere University

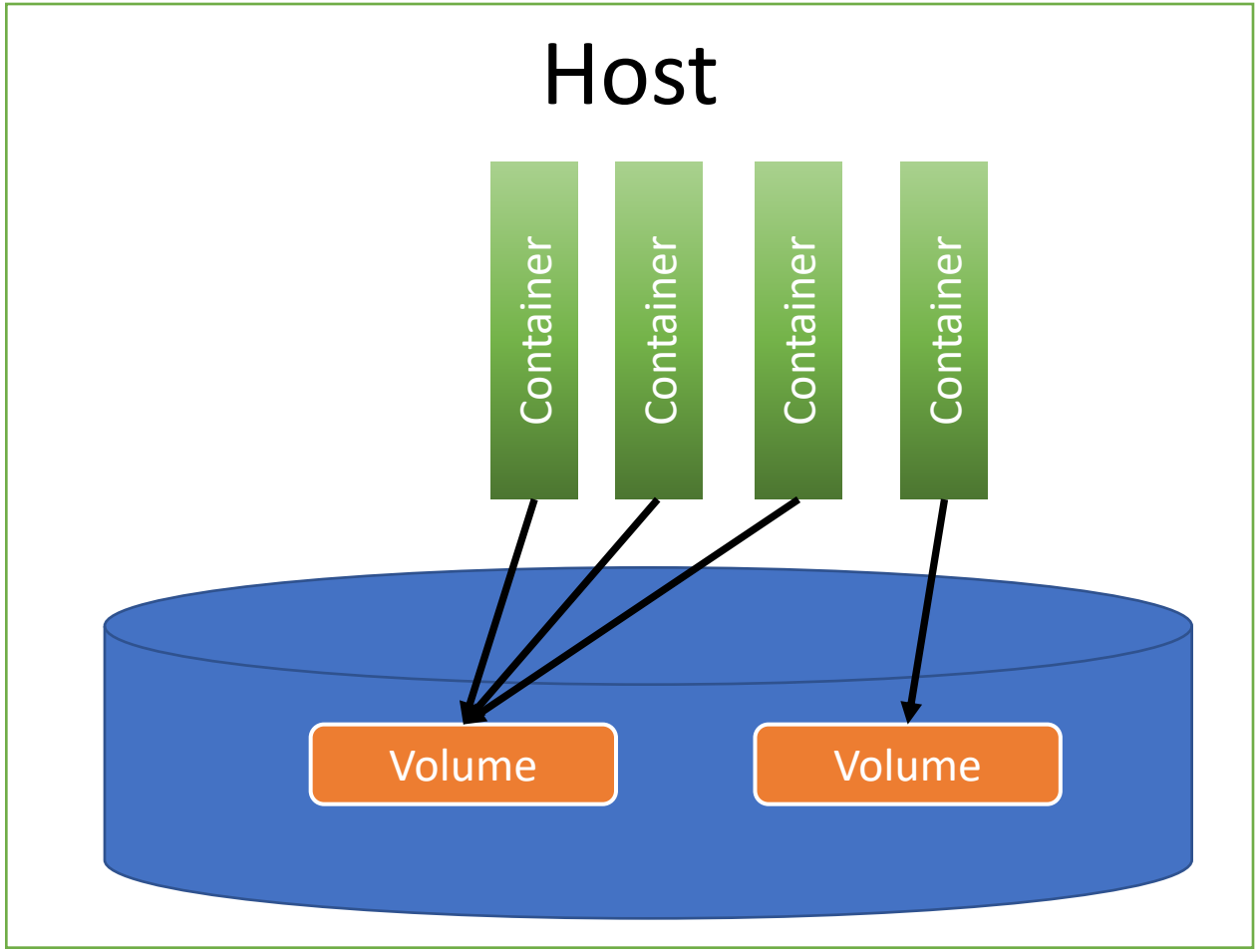
Typical set-up



Tampereen yliopisto
Tampere University

Typical set-up

Volumes are for
- Persistent data
- Data sharing



Tampereen yliopisto
Tampere University

Dockerfile

```
#This is a sample Image
FROM ubuntu
MAINTAINER demousr@gmail.com
RUN apt-get update
RUN apt-get install -y nginx
CMD [“echo”,”Image created”]
```

Tampereen yliopisto
Tampere University

- `$ docker image history 874176ca6a7c`

```
IMAGE          CREATED         CREATED BY                                      SIZE COMMENT
874176ca6a7c 6 months ago  /bin/sh -c #(nop)  CMD  ["/bin/sh" "-c" "nodeâ¦     0B
e477afdf62b1 6 months ago  /bin/sh -c #(nop)  EXPOSE  8893                     0B
bbb82ecea436 6 months ago  /bin/sh -c #(nop)  ENV PORT=8893                    0B
f7e49216f8f8 6 months ago  /bin/sh -c #(nop)  WORKDIR /home                    0B
75bf5968b264 6 months ago  /bin/sh -c #(nop)  COPY  dir:fed108fd8e77a8ed8â¦ 553kB
1fb4fbe8beb7 7 months ago  /bin/sh -c #(nop)  USER root                        0B
8232a8b9c483 2 years ago   /bin/sh -c #(nop)  CMD  ["node"]                    0B
<missing>    2 years ago   /bin/sh -c apk add --no-cache  --virtual .buiâ¦  3.56MB
<missing>    2 years ago   /bin/sh -c #(nop) ENV YARN_VERSION=0.21.3          0B
<missing>    2 years ago   /bin/sh -c adduser -D -u 1000  node && apâ¦      45.4MB
<missing>    2 years ago   /bin/sh -c #(nop) ENV NODE_VERSION=6.10.0          0B
<missing>    2 years ago   /bin/sh -c #(nop) ENV NPM_CONFIG_LOGLEVEL=iâ¦      0B
<missing>    2 years ago   /bin/sh -c #(nop) ADD file:3df55c321c1c8d73fâ¦   4.81MB
```

```
FROM node:6.10.0-alpine
# Never run processes as root!
USER root

# Copy application itself:
COPY . /home

WORKDIR /home

# Set port on …:
ENV PORT=8893

# Expose port 8893:
EXPOSE 8893

CMD node pinger.js
```

A layer describes a difference to the previous layer.

Layers of size 0 are ”intermediate” layers

Non-zero layers are created with RUN, COPY and ADD.

Tampereen yliopisto
Tampere University

Schedule for coming weeks

Week	Lecture	Plussa exercises (deadlines)
-1		
0		17.08 Plussa open for students
1 / 35	25.08 Intro to the course and topic.	20.08 Background survey opens
2 / 36	01.09 Virtualization, what, why and how. Intro to containers and Docker.	04.09 Background survey closes
3 / 37	08.09 Cloud and scalability, implications to SW development and business	11.09 Docker exercise closes
4 / 38	15.09 Continuous deployment, what and why	14.09 Docker compose e. opens
5 / 39	21.09 Continuous deployment, tools and techniques	
6 / 40	28.09 Issues on cloud-SW: isolation, dependency management etc	01.10 Docker compose e. closes