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	11.09/17.09 Docker exercise closes
4 / 38	15.09 Continuous deployment, what and why	17.09 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

Tampereen yliopisto
Tampere University

About you

- Plus.tuni.fi has now 92 enrolled students
- 67 of you answered to background questonary
  - Less than half meet the formal pre-requisities,
    but about 50 students seem to have otherwise strong background, and
    17 might have taken this course too early.

- I still hope that clearly over 50 students finally pass the course

Tampereen yliopisto
Tampere University

Next exercise

In this exercise you will create a simple docker file and run in.

Your task is to

- implement a simple “Hello, World” application in any programming language you wish. I hope to see many!
- Create Dockerfile that runs this application (need to have a compiler or runtime for the selected programming language)
- Build the docker image and run it.

Your return should include three things

- Content of the Dockerfile
- Output of “docker history” of your created image
- Source code of the application

In addition we ask you to answer couple questions. These answers are not graded, but they are used to develop this course.

**DEADLINE FOR FULL POINTS IS 11.09.2020. THIS SECTION CLOSES 17.09.2020. **

Links to Docker material can be found from <https://plus.tuni.fi/comp.se.140/fall-2020/c01_intro/03_material/>

Tampereen yliopisto
Tampere University

“Kooditorio”

- In normal times we have a special place where students can come with their problems.
- Now, we try that with zoom-sessions at 1400-1530.

Couple of job adds

- Patria is looking for an SW engineer
  https://pm-careers.rekrytointi.com/paikat/?o=A_RJ&jgid=1&jid=1508
- Hitachi ABB Power Grids
  Looks for part time (18h/w) employee
- Our VISDOM project (https://iteavisdom.org)
  Can hire a master thesis worker.

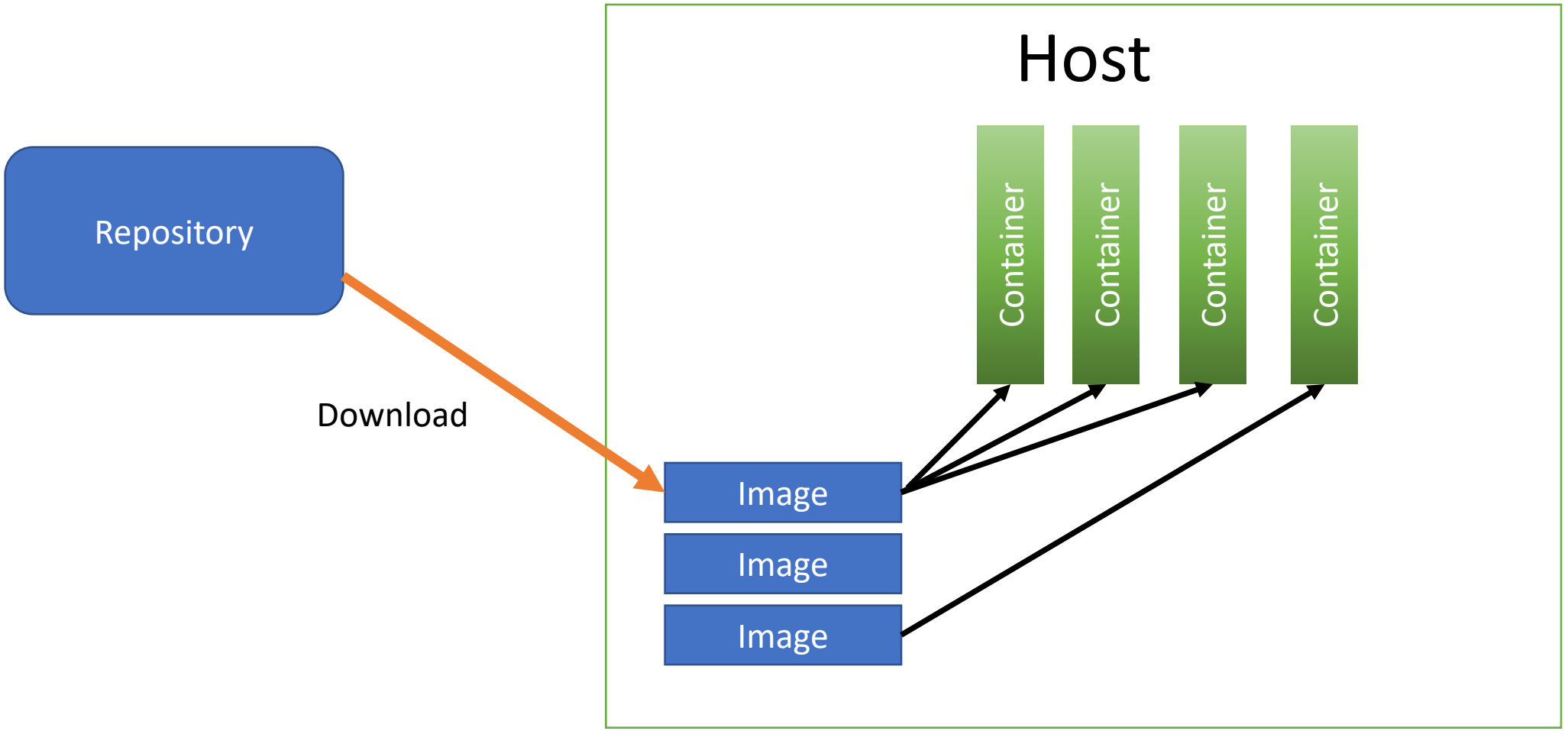
- Contact me (kari.systa@tuni.fi) if you are interested

Tampereen yliopisto
Tampere University

Recap on Docker

Tampereen yliopisto
Tampere University

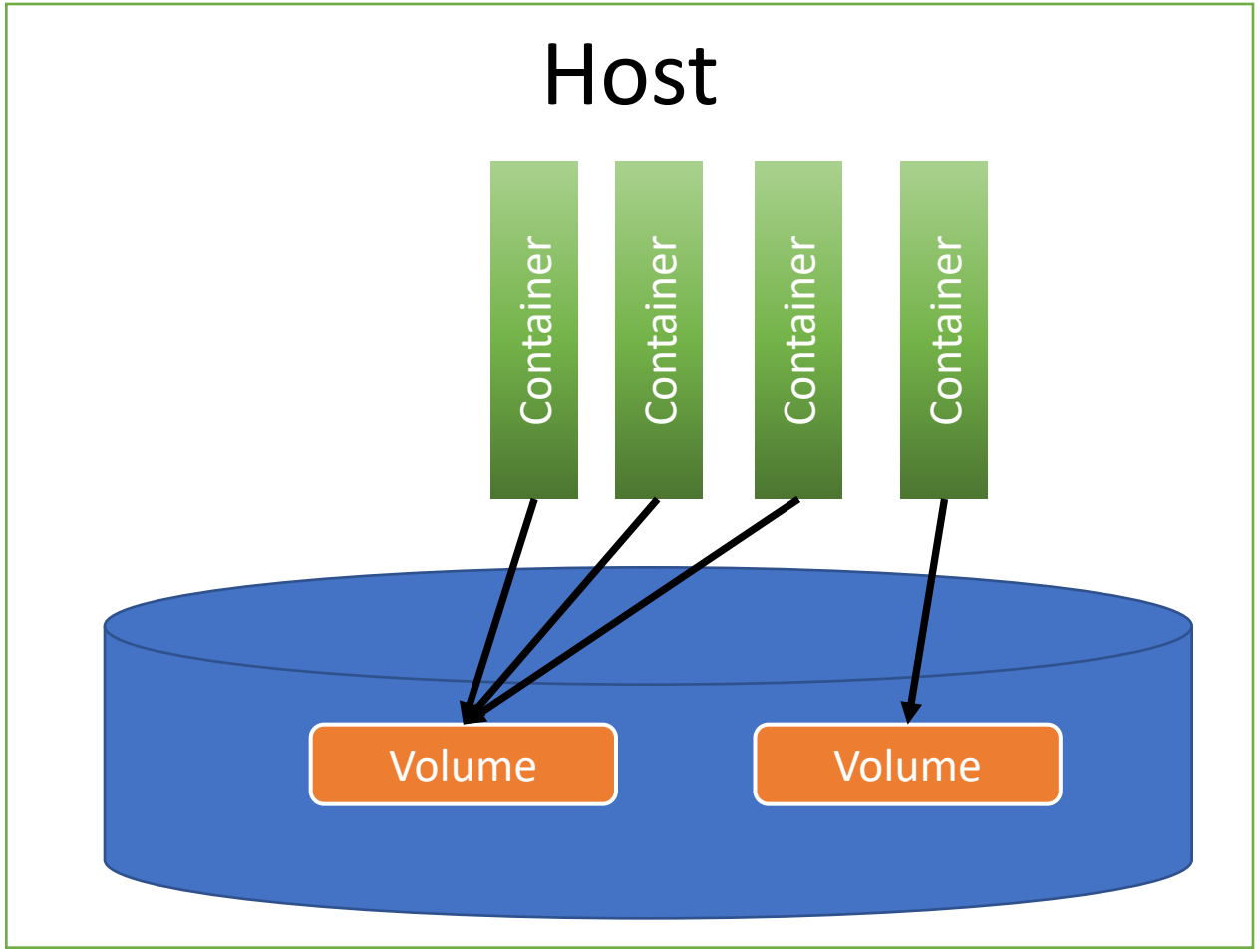
Typical set-up



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 commands (subset of)

docker build	Build an image from a Dockerfile
docker container	Manage containers
docker commit	Create a new image from a container's changes
docker exec	Run a command in a running container
docker image	Manage images
docker inspect	Return low-level information on Docker objects
docker ps	List containers
docker run	Run a command in a new container
docker stop	Stop one or more running containers
docker swarm	Manage Swarm
docker volume	Manage volumes

Tampereen yliopisto
Tampere University

```
kari@lepikko-nuc:~$ docker image ls
REPOSITORY                TAG                 IMAGE ID            CREATED             SIZE
rabbitmq                  3-management        38e57f281891        5 months ago        184MB
rabbitmq                  3                   ce51f7cc8a59        5 months ago        154MB
<none>                    <none>              68ea2e469580        10 months ago       69.2MB
python                    3.6                 a2e9f0fba405        10 months ago       913MB
<none>                    <none>              f49eea6b572d        11 months ago       69.2MB
apluslms/grading-base     latest              669ef3db6a57        11 months ago       160MB
debian                    buster-20190812-slim 83a10817c894       12 months ago       69.2MB
gitlab/gitlab-runner      latest              53b7cdd9ce4a        18 months ago       408MB
gitlab/gitlab-ce          latest              8be4d926d44e        18 months ago       1.62GB
pinger                    latest              874176ca6a7c        18 months ago       54.4MB
composetest_pingrelay     latest              f1915b52acde        18 months ago       54.4MB
<none>                    <none>              653b72c389fe        18 months ago       54.4MB
<none>                    <none>              0ceea4e0dde7        18 months ago       53.8MB
<none>                    <none>              2a1516bbcf7c        18 months ago       53.8MB
<none>                    <none>              6f8e8a11dded        18 months ago       53.8MB
<none>                    <none>              ce0842e65950        18 months ago       53.8MB
<none>                    <none>              5c3f5be3e518        18 months ago       53.8MB
<none>                    <none>              1037fb9e5f21        18 months ago       53.8MB
<none>                    <none>              840f0aae6c3d        18 months ago       54.4MB
<none>                    <none>              b0defb476587        18 months ago       54.4MB
<none>                    <none>              3299381687af        18 months ago       54.4MB
<none>                    <none>              0eafd852183d        18 months ago       54.4MB
<none>                    <none>              08656cd757e0        18 months ago       54.4MB
<none>                    <none>              2294d910c093        18 months ago       54.4MB
<none>                    <none>              dd4036ed33e0        18 months ago       53.8MB
pingrelay                 latest              52b9f6060548        18 months ago       53.8MB
<none>                    <none>              68f5bdacbb58        18 months ago       53.8MB
<none>                    <none>              b1b59d44b5fb        22 months ago       2.61GB
liquidiot-docker_ide      latest              79214e5a582a        22 months ago       682MB
liquid-ide2               latest              c52aeb0af6ce        22 months ago       2.28GB
liquidiot-ide             latest              b20073f68dcd        22 months ago       1.67GB
liquidiot-base            latest              b88e10ba3196        22 months ago       205MB
hellotest                 latest              27c7e8db4dd6        22 months ago       132MB
friendlyhello             latest              dfd0ebf0f838        22 months ago       132MB
ubuntu                    14.04               f216cfb59484        22 months ago       188MB
ubuntu                    latest              ea4c82dcd15a        22 months ago       85.8MB
python                    2.7-slim            804b0a01ea83        22 months ago       120MB
mongo                     3.2.21              0d365aaccf85        22 months ago       300MB
hello-world               latest              4ab4c602aa5e        24 months ago       1.84kB
piehei/base-qt593         latest              4318f4a85238        2 years ago         10.3GB
node                      6.10.0-alpine       8232a8b9c483        3 years ago         53.8MB
```

Tampereen yliopisto
Tampere University

```
kari@lepikko-nuc:~$ docker container ls
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS              PORTS                    NAMES
b74ff2393ba3        gitlab/gitlab-runner:latest    "/usr/bin/dumb-init …"   10 months ago       Up 29 minutes       0.0.0.0:8000->80/tcp     xxx2
a1830c0f526f        gitlab/gitlab-runner:latest    "/usr/bin/dumb-init …"   10 months ago       Up 29 minutes                                gitlab-runner
```

Tampereen yliopisto
Tampere University

```
FROM node:6.10.0-alpine
USER root
COPY . /home
WORKDIR /home
ENV PORT=8893
EXPOSE 8893

CMD node pinger.js
```

$ docker history 874176ca6a7c

IMAGE	CREATED	CREATED BY	SIZE
874176ca6a7c	18 months ago	/bin/sh -c #(nop) CMD ["/bin/sh" "-c" "node…	0B
e477afdf62b1	18 months ago	/bin/sh -c #(nop) EXPOSE 8893	0B
bbb82ecea436	18 months ago	/bin/sh -c #(nop) ENV PORT=8893	0B
f7e49216f8f8	18 months ago	/bin/sh -c #(nop) WORKDIR /home	0B
75bf5968b264	18 months ago	/bin/sh -c #(nop) COPY dir:fed108fd8e77a8ed8…	553kB
1fb4fbe8beb7	18 months ago	/bin/sh -c #(nop) USER root	0B
8232a8b9c483	3 years ago	/bin/sh -c #(nop) CMD ["node"]	0B
<missing>	3 years ago	/bin/sh -c apk add --no-cache --virtual .bui…	3.56MB
<missing>	3 years ago	/bin/sh -c #(nop) ENV YARN_VERSION=0.21.3	0B
<missing>	3 years ago	/bin/sh -c adduser -D -u 1000 node && ap…	45.4MB
<missing>	3 years ago	/bin/sh -c #(nop) ENV NODE_VERSION=6.10.0	0B
<missing>	3 years ago	/bin/sh -c #(nop) ENV NPM_CONFIG_LOGLEVEL=i…	0B
<missing>	3 years ago	/bin/sh -c #(nop) ADD file:3df55c321c1c8d73f…	4.81MB

Tampereen yliopisto
Tampere University

Some Dockerfile commands

- FROM <image> [AS <name>]
- RUN <command>
  execute any commands in a new layer and commit the results.
- CMD
  default command to be executed when the container starts
- ENV <key> <value>
- ADD [--chown=<user>:<group>] <src>... <dest>
- COPY [--chown=<user>:<group>] <src>... <dest>

- **kari@lepikko-nuc**:**~/CloudApplications/test**$ docker build .
- Sending build context to Docker daemon 3.072kB
- Step 1/3 : FROM debian:buster-20190812-slim
- ---> 83a10817c894
- Step 2/3 : RUN date > /home/123456.txt
- ---> Using cache
- ---> 38b806d0d640
- Step 3/3 : CMD cat /home/123456.txt
- ---> Using cache
- ---> 68ea2e469580
- Successfully built 68ea2e469580
- **kari@lepikko-nuc**:**~/CloudApplications/test**$ docker run 68ea2e469580
- Wed Oct 23 16:34:26 UTC 2019

kari@lepikko-nuc:~/CloudApplications/test$ docker inspect 68ea2e469580

[
    {
        "Id":
"sha256:68ea2e469580458998dfdc9c0a13db39541803e6f988245ee55f2b124fb1035f",
        "RepoTags": [],
        "RepoDigests": [],
        "Parent":
"sha256:38b806d0d64026fe73682cead45e908977261839 2a86fe29cff70f0aebff9a2c",
        "Comment": "",
        "Created": "2019-10-23T16:34:27.518846277Z",

kari@lepikko-nuc:**~/CloudApplications/test**$ docker image rm 68ea2e469580

Error response from daemon: conflict: unable to delete 68ea2e469580 (must be forced) - image is being used by stopped container 832701d84725

kari@lepikko-nuc:**~/CloudApplications/test**$ docker image rm -f 68ea2e469580

Deleted: sha256:68ea2e469580458998dfdc9c0a13db39541803e6f988245ee55f2b124fb1035f

Deleted: sha256:38b806d0d64026fe73682cead45e908977261839 2a86fe29cff70f0aebff9a2c

```
kari@lepikko-nuc:~/CloudApplications/test$ docker build .
Sending build context to Docker daemon   5.12kB
Step 1/3 : FROM debian:buster-20190812-slim
 ---> 83a10817c894
Step 2/3 : RUN date > /home/123456.txt
 ---> Running in 646f50d34715
Removing intermediate container 646f50d34715
 ---> d97b91c429fc
Step 3/3 : CMD cat /home/123456.txt
 ---> Running in 2750a97284f8
Removing intermediate container 2750a97284f8
 ---> fc4c5d3376a9
Successfully built fc4c5d3376a9
kari@lepikko-nuc:~/CloudApplications/test$ docker run fc4c5d3376a9
Sun Sep  6 08:53:24 UTC 2020
```

Tampereen yliopisto
Tampere University

```
kari@lepikko-nuc:~/CloudApplications/test$ docker history fc4c5d3376a9
IMAGE               CREATED             CREATED BY                                      SIZE
fc4c5d3376a9        About a minute ago  /bin/sh -c #(nop)  CMD ["/bin/sh" "-c" "cat …   0B
d97b91c429fc        About a minute ago  /bin/sh -c date > /home/123456.txt              29B
83a10817c894        12 months ago       /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>           12 months ago       /bin/sh -c #(nop) ADD file:330bfb91168adb4a9…   69.2MB
kari@lepikko-nuc:~/CloudApplications/test$
```

Docker image ls

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
<none>              <none>              99d4b9dd0fea        32 seconds ago      1.19GB
<none>              <none>              f86de57f2f64        2 minutes ago       1.19GB
<none>              <none>              c6f93e248201        14 minutes ago      944MB
<none>              <none>              8d4c9c98fece        15 minutes ago      944MB
<none>              <none>              51257e92bd03        18 minutes ago      944MB
<none>              <none>              fc4c5d3376a9        33 minutes ago      69.2MB
node                latest              40ce906a3734        4 days ago          944MB
gcc                 latest              cfe277915109        4 days ago          1.19GB
rabbitmq            3-management        38e57f281891        5 months ago        184MB
rabbitmq            3                   ce51f7cc8a59        5 months ago        154MB
```

kari@lepikko-nuc:~/CloudApplications/DockerExercise$ docker build -f Dockerfile.gcc -t gcc-hello .

```
Successfully built 99d4b9dd0fea
Successfully tagged gcc-hello:latest
```

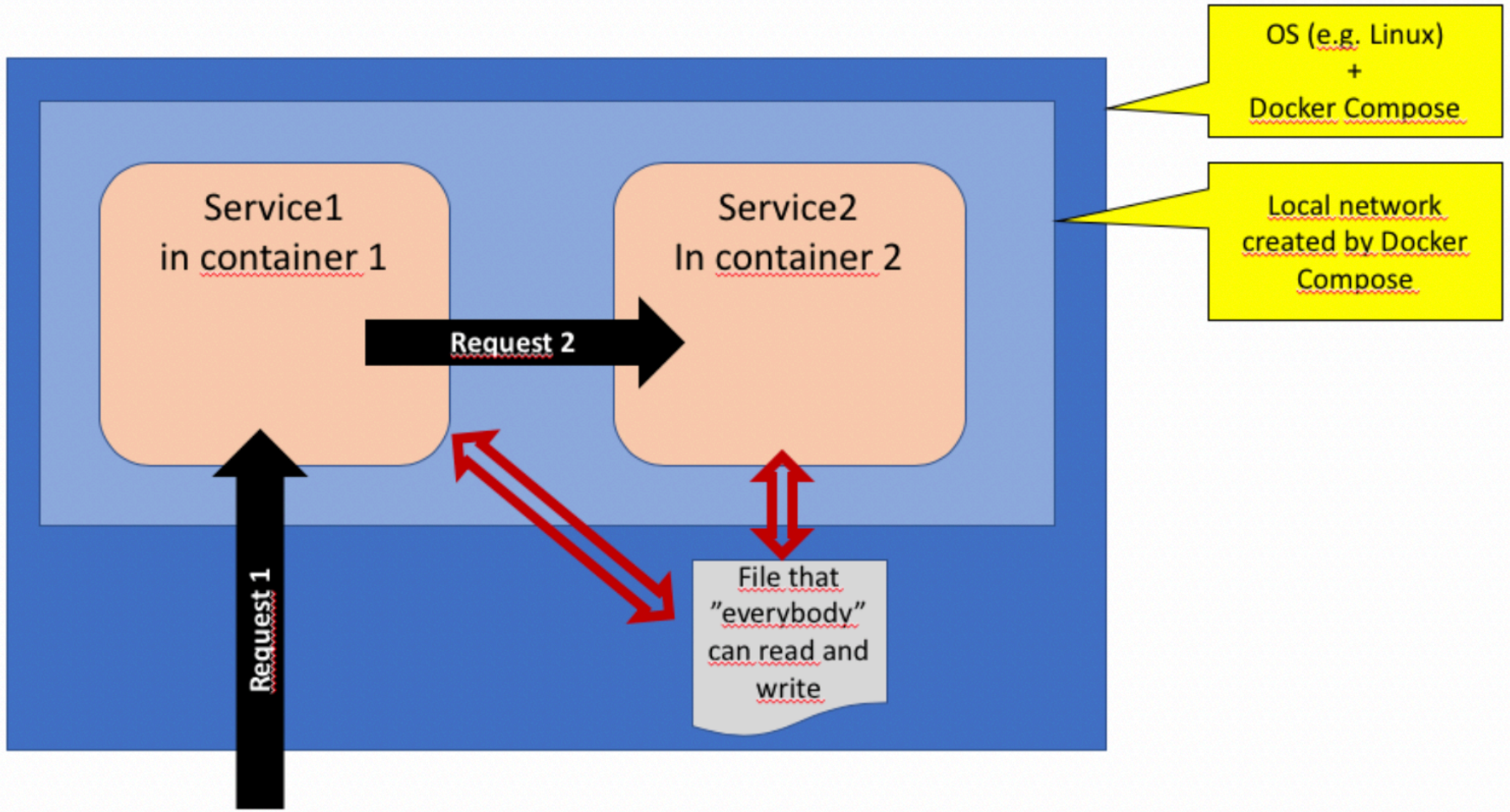
kari@lepikko-nuc:~/CloudApplications/DockerExercise$ docker image ls

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
gcc-hello           latest              99d4b9dd0fea        28 minutes ago      1.19GB
<none>              <none>              f86de57f2f64        30 minutes ago      1.19GB
<none>              <none>              c6f93e248201        42 minutes ago      944MB
```

Tampereen yliopisto
Tampere University

A sneak peak to docker-compose

- A technology to set up a group of interworking containers
- Configuration defined in file ’docker-compose.yaml’



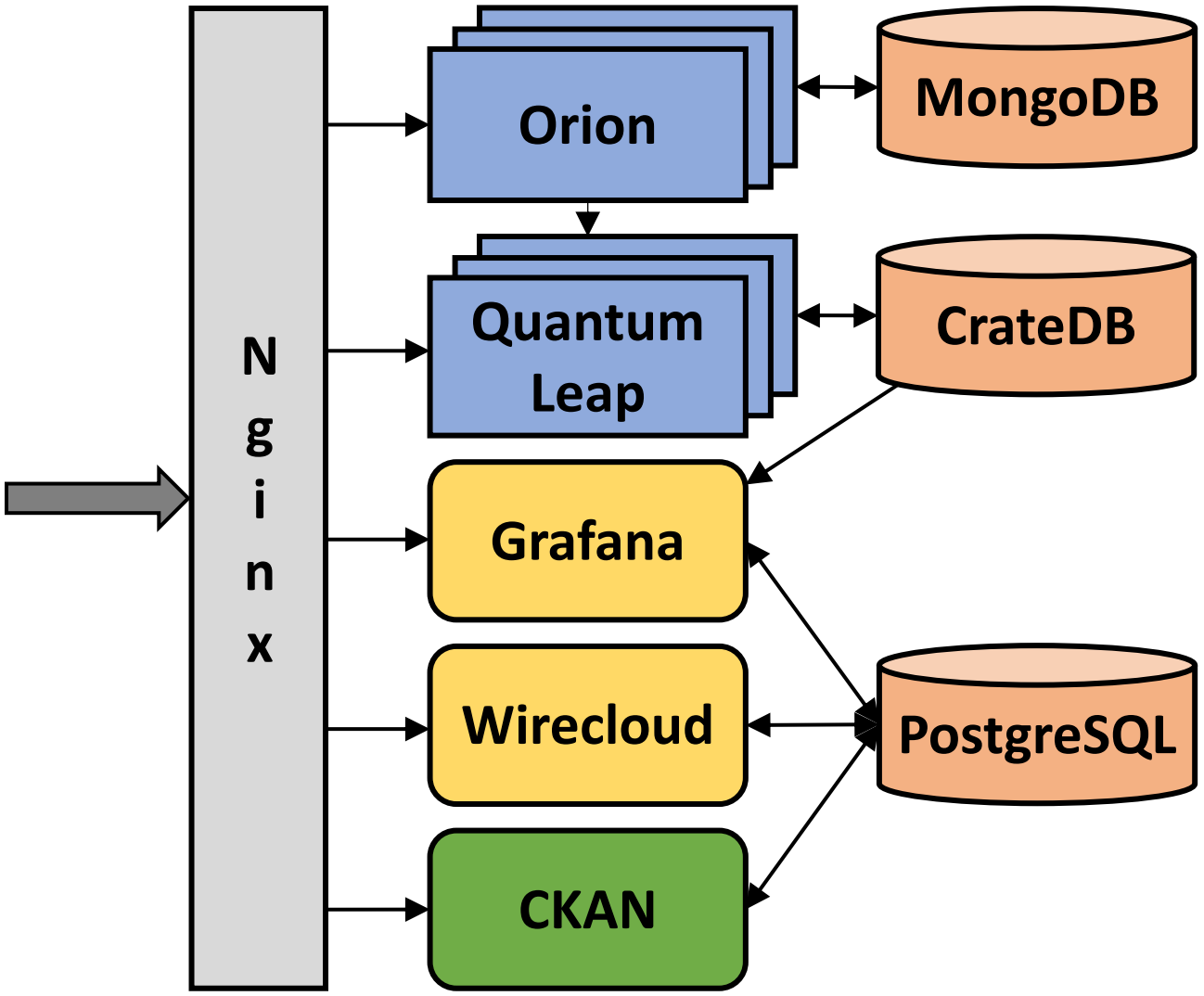

```
version: '3'

services:

  pinger:
     image: "pinger”
     ports:
       - "8893:8893”
     networks:
       - pingnet
   volumes:
       - ./data:/data
     environment:
       ServiceName: service_2

  pingrelay:
     build: "pingrelay”
     ports:
       - "8004:8894”
     networks:
       - pingnet
     volumes:
       - ./data:/data
     environment:
       ServiceName: service_1

networks:
    pingnet:

volumes:

  data: {}
```

Tampereen yliopisto
Tampere University

Docker Swarm

- Clustering for scalability

FIWARE platform architecture



FIWARE Core Component

Dashboard Component

Data Management Component

Database

Access control, proxy server

FIWARE access control components (Keyrock, Wilma and AuthZForce) are not included in this document.

Vagrant

Tampereen yliopisto
Tampere University

Vagrant intro

- A way to create and distribute development environments as virtual machine (full VMs – not containers)
- If time lets look: https://www.vagrantup.com/intro/index.html

Tampereen yliopisto
Tampere University

Vagrant vs Docker (https://www.vagrantup.com/intro/vs/docker.html)

- Vagrant is a tool focused on providing a consistent development environment workflow across multiple operating systems. Docker is a container management that can consistently run software as long as a containerization system exists.
- Containers are generally more lightweight than virtual machines, so starting and stopping containers is extremely fast. Docker uses the native containerization functionality on macOS, Linux, and Windows.
- Currently, Docker lacks support for certain operating systems (such as BSD). If your target deployment is one of these operating systems, Docker will not provide the same production parity as a tool like Vagrant. Vagrant will allow you to run a Windows development environment on Mac or Linux, as well.
- For microservice heavy environments, Docker can be attractive because you can easily start a single Docker VM and start many containers above that very quickly. This is a good use case for Docker. Vagrant can do this as well with the Docker provider. A primary benefit for Vagrant is a consistent workflow but there are many cases where a pure-Docker workflow does make sense.
- Both Vagrant and Docker have a vast library of community-contributed "images" or "boxes" to choose from.

Tampereen yliopisto
Tampere University

Introduction to cloud (Back to general concepts)

Tampereen yliopisto
Tampere University

Agenda

- History and motivations
- Definitions
- Categories: IaaS, SaaS, PaaS
- Business and SW development perpective
- (Technical perspectives in Cloud Platforms course)

Tampereen yliopisto
Tampere University

Our perception?

Different interpretations

- It is somewhere in the cloud and controlled by a ”big brother”
- We do not need to buy our computing hardware anymore
- Cool technology to play with
- Accessible from anywhere
- Other

Tampereen yliopisto
Tampere University

Where did it start ?

https://www.dataversity.net/brief-history-cloud-computing

(Virtualization is an old invention)

(We discussed it a week ago)

- ” In its early stages, the Cloud was used to express the empty space between the end user and the provider. In 1997, Professor Ramnath Chellapa of Emory University defined Cloud Computing as the new *’computing paradigm, where the boundaries of computing will be determined by economic rationale, rather than technical limits alone.’* ”
- In 1999, Salesforce became a popular example of using Cloud Computing successfully. They used it to pioneer the idea of using the Internet to deliver software programs to the end users. The program (or application) could be accessed and downloaded by anyone with Internet access. Businesses could purchase the software in an on-demand, cost-effective manner, without leaving the office.

Tampereen yliopisto
Tampere University

Case salesforce

- From https://www.salesforce.com/:

WHAT IS SALESFORCE?

Salesforce is a suite of web-based CRM applications that help you find, win, and keep customers. Learn more.

Tampereen yliopisto
Tampere University

https://www.computerworld.com/article/3427741/a-brief-history-of-salesforce-com.html

“The way the story goes is that Marc Benioff was floating in the sea just off Big Island in his beloved Hawaii during a sabbatical when he thought: **why can't buying software be as a simple as Amazon is for consumer goods**?

This line of thinking eventually led to Benioff and a team of developers pioneering the software-as-a-service (SaaS) model by delivering its customer relationship management (CRM) software **over the internet on a per seat, per month payment plan, instead of deployed on-premise servers under a hefty licensing agreement**.”

Tampereen yliopisto
Tampere University

Taivalsaari, Systä: Cloudberry: An HTML5 Cloud Phone Platform for Mobile Devices IEEE Software July/August 2012

A research prototype in which the application stack of a smart phone was implemened with HTML5.

Key components of Cloudberry

- Modern browser engine
- Downloadable top-level Uis
- Suite of web applications
- Data API that transparently stored to backend
- Set of device APIs
- Domain and permission based security model
- Process model

Tampereen yliopisto
Tampere University

For more information see the article

FOCUS: MOBILE SOFTWARE DEVELOPMENT

Cloudberry:
An HTML5 Cloud Phone Platform for Mobile Devices

Antero Taivalsaari and Kari Systä, Nokia

// One of the central benefits of a cloud phone is that nearly any of its customer-facing features can be changed on millions of devices all over the world almost instantly. //

customize devices to different users and purposes. Another key benefit is multiple device ownership—that is, the ability to effortlessly access the same applications and data from different devices.

The Cloud Phone

A cloud phone is a mobile device in which all customer-facing functionality is downloaded and cached dynamically from the Web, including all the applications and even the entire top-level user interface (UI) of the device. For more information on mobile cloud systems, see the "Related Work in Mobile Cloud Systems" sidebar. A cloud phone will have several key characteristics.

Tampereen yliopisto
Tampere University

Case Amazon

- Forerunner of web-based retail services.
- Used only 10% of their computing capacity capacity (which was commonplace at the time).
- The Cloud Computing Infrastructure Model gave them the flexibility to use their computer's capacity much more efficiently.
- Soon after, other large organizations followed their example.
- In 2006, Amazon launched Amazon Web Services, which offers online services to other websites, or clients.
- Another of Amazon Web Services' sites is the Elastic Compute Cloud (EC2), allowing individuals to rent virtual computers and use their own programs and applications.
- Now selling computing capacity to others (AWS business) brings about half of the operating income (source: https://press.aboutamazon.com/news-releases/news-release-details/amazoncom-announces-first-quarter-sales-17-597-billion)

Tampereen yliopisto
Tampere University

Case Google

- Also in 2006, Google launched the Google Docs services.
- Based on two separate products,
  - Google Spreadsheets (acquired from 2Web Technologies, in 2005) and
  - Writely (Google purchased Writely)
- In 2007, IBM, Google, and several universities joined forces to develop a server farm for research projects needing both fast processors and huge data sets.
- 2007 was also the year when Netflix launched it’s streaming video service, using the Cloud, and provided support for the practice of “binge-watching.”

Tampereen yliopisto
Tampere University

Story continues

- IBM
  - Smart Cloud
  - IBM Cloud (formerly IBM Bluemix and IBM SoftLayer)
- Microsoft
  - Azure
  - Office 360
- Apple
  - ICloud

Tampereen yliopisto
Tampere University

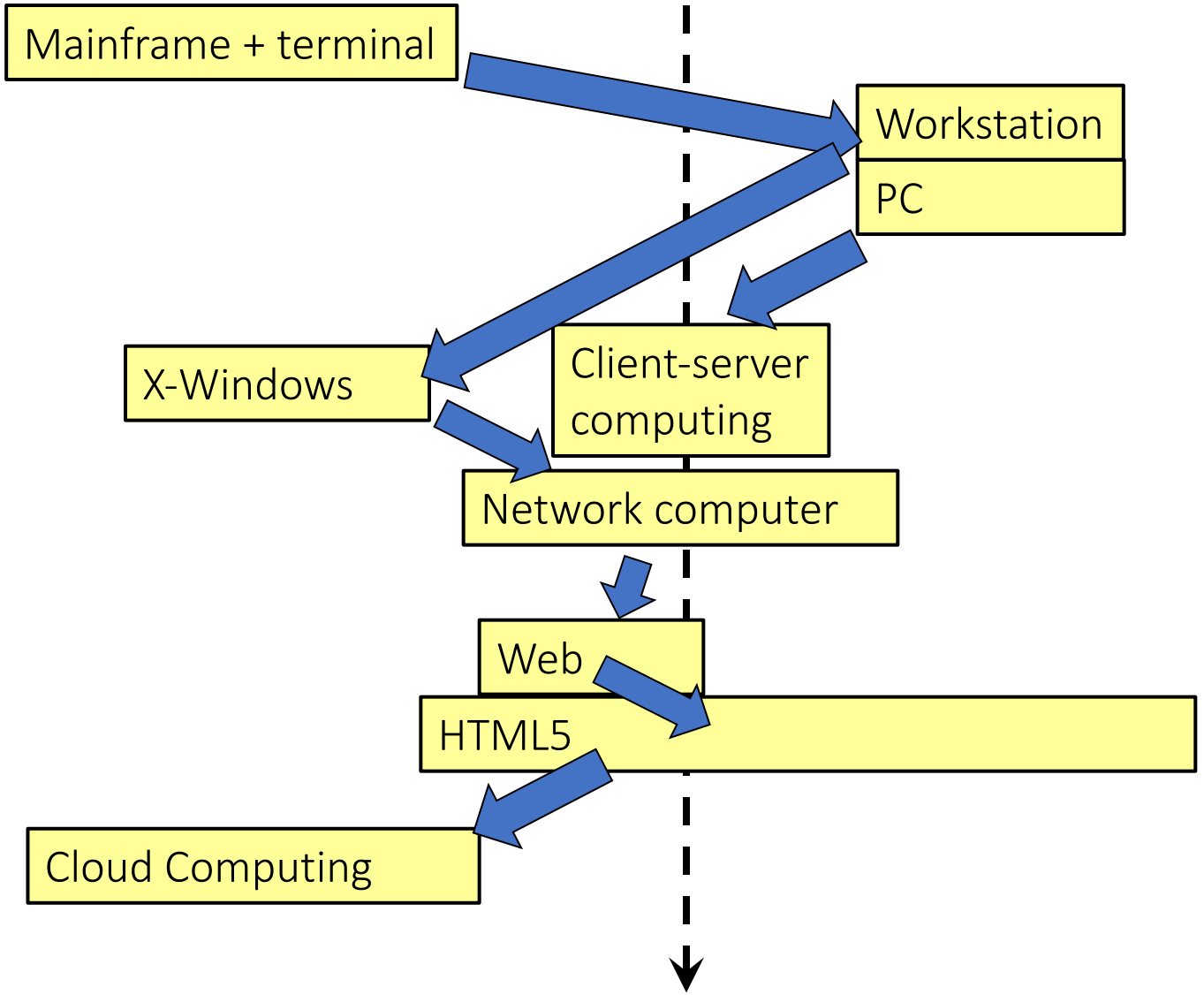
Example SaaS: Adobe Creative Cloud

(http://www.paulpehrson.com/2011/04/11/adobes-new-software-as-a-service-model/)

Product	Full	Upgrade*	SAAS**	Months to justify initial investment***
Design Premium	$1899	$399	$95	20
Web Premium	$1799	$399	$89	20
Production Premium	$1699	$399	$85	20
Master Collection	$2599	$549	$129	20
Photoshop	$699	$199	$35	20
Illustrator	$599	$199	$29	20

Tampereen yliopisto
Tampere University

Evolution of client-server split (thin vs thick client)



Tampereen yliopisto
Tampere University

Towards definitions

Peter Mell; Timothy Grance (September 2011). The NIST Definition of Cloud Computing (Technical report). National Institute of Standards and Technology: U.S. Department of Commerce. doi:10.6028/NIST.SP.800-145. Special publication 800-145.

Tampereen yliopisto
Tampere University

Essential characteristics 1/2

- *On-demand self-service.* A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
  - Unilaterally?
  - Without human interaction?
- *Broad network access.* Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
  - What does the heterogeneous platforms mean in practice?
- *Rapid elasticity.* Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
  - What if scaling is not automatic?

Tampereen yliopisto
Tampere University

Essential characteristics 2/2

- *Resource pooling.* The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.
  - Why is this essential?
- *Measured service.* Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.
  - Why?

Tampereen yliopisto
Tampere University

Service models

SaaS

PaaS

IaaS

Infrastructure as a Service (IaaS). The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

Examples?

Tampereen yliopisto
Tampere University

Service models

SaaS

PaaS

IaaS

Platform as a Service (PaaS). The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

Examples?

Tampereen yliopisto
Tampere University

Service models

SaaS

PaaS

IaaS

Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Examples?

Tampereen yliopisto
Tampere University

The examples are discussed next week

Tampereen yliopisto
Tampere University

Deployment models

- *Private cloud.* The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.
- *Community cloud.* The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.
- *Public cloud.* The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.
- *Hybrid cloud*. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).