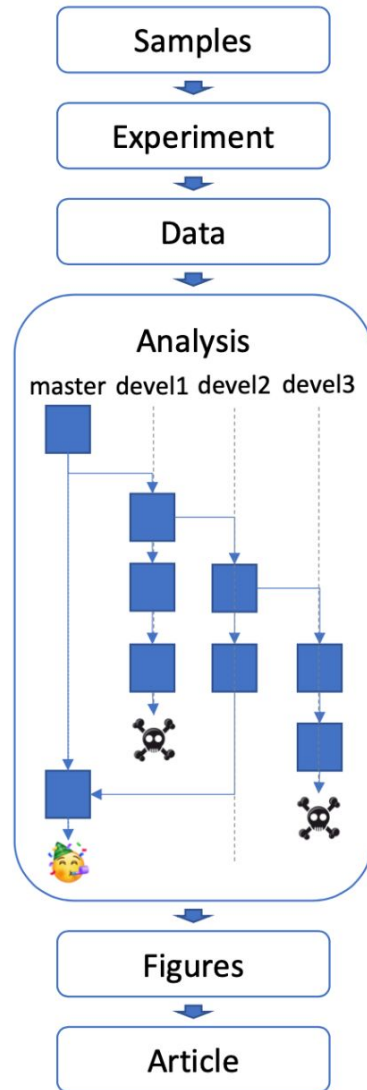


Use case 2 : Encapsulation

Céline Hernandez, I2BC, 0000-0001-8664-1340

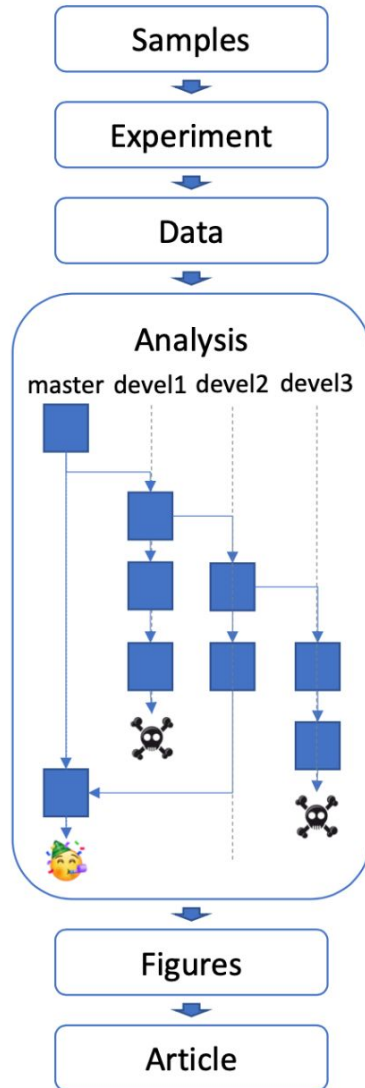


A (not-so-uncommon) nightmare





A (not-so-uncommon) nightmare

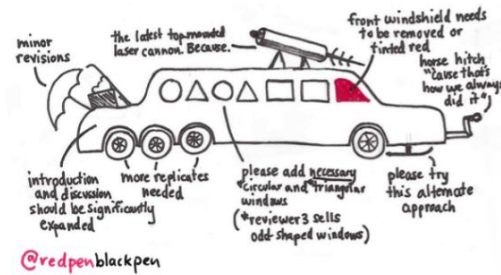


Submit to a journal

Reviewer 1



Reviewer 2

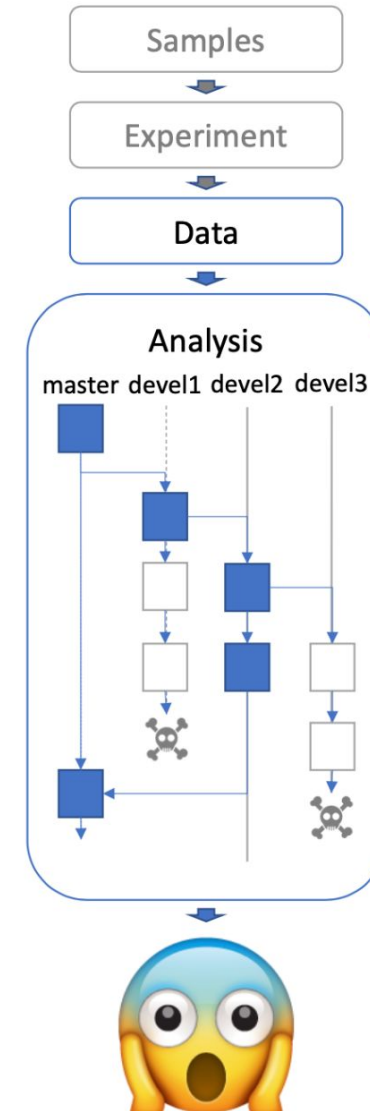
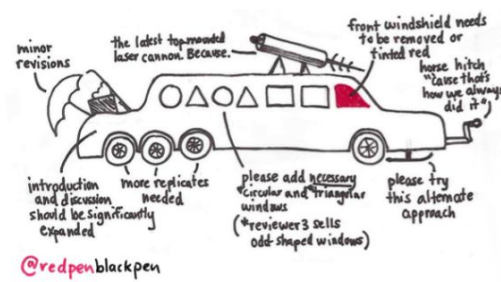
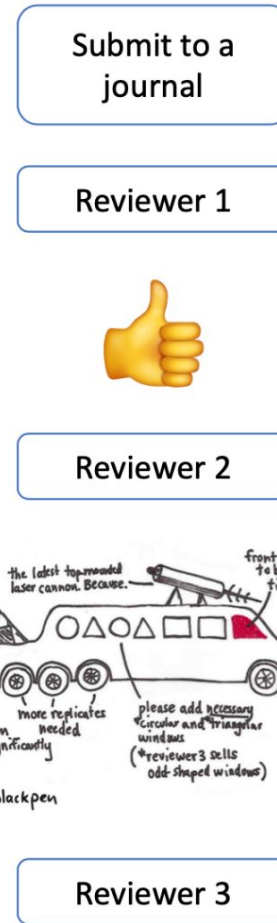
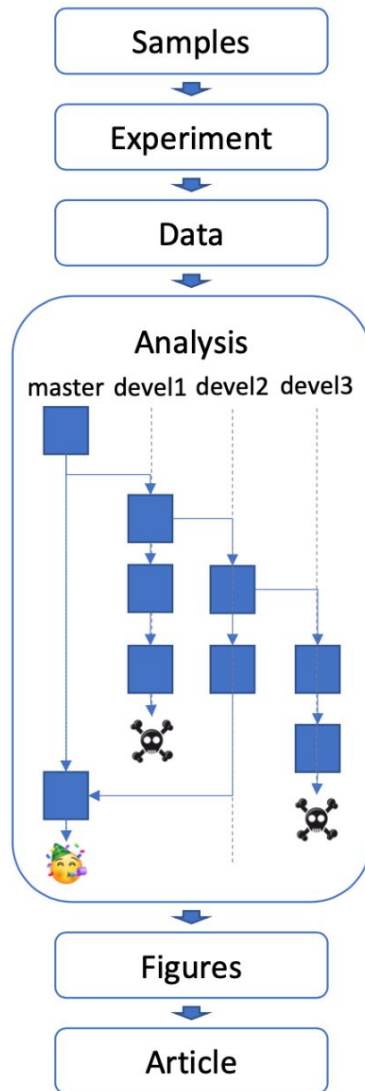


Reviewer 3



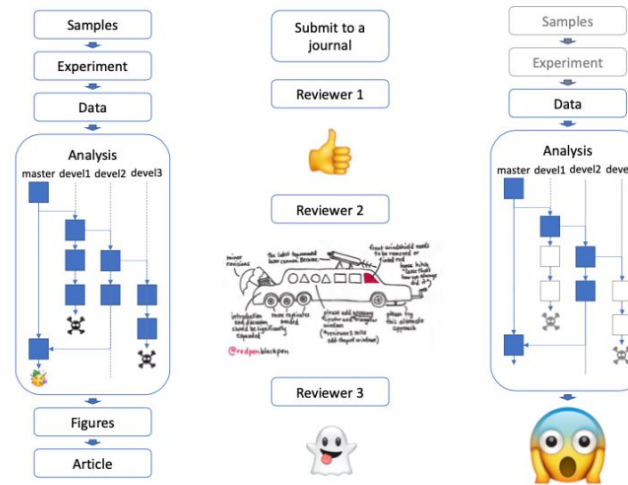


A (not-so-uncommon) nightmare





A (not-so-uncommon) nightmare






What changed?

- Package
- Software
- Libraries
- Environment variables
- OS version
- Computer
- ..?



Goal : capture the system environment of applications (OS, packages, libraries,...) to control their execution.

- Hardware virtualisation (virtual machines) 
- OS virtualisation (images and containers) 
- Environment management 



Let's say we want to install RStudio...

Install Rstudio ?



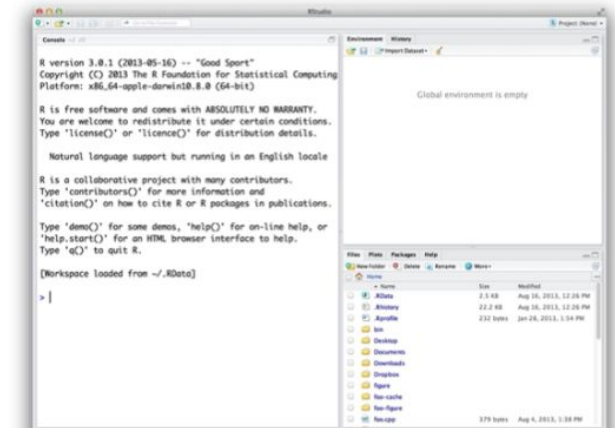
MacOS



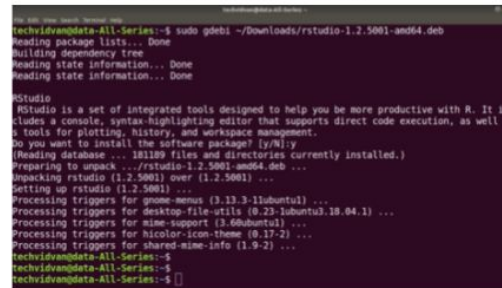
Windows



Use Rstudio

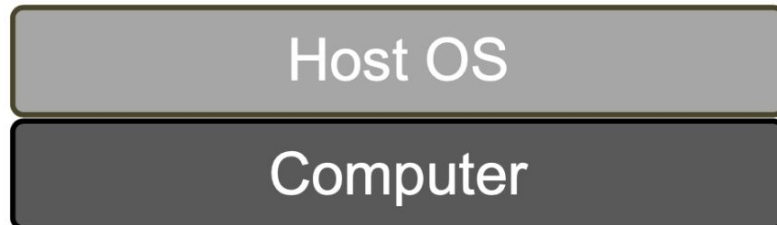


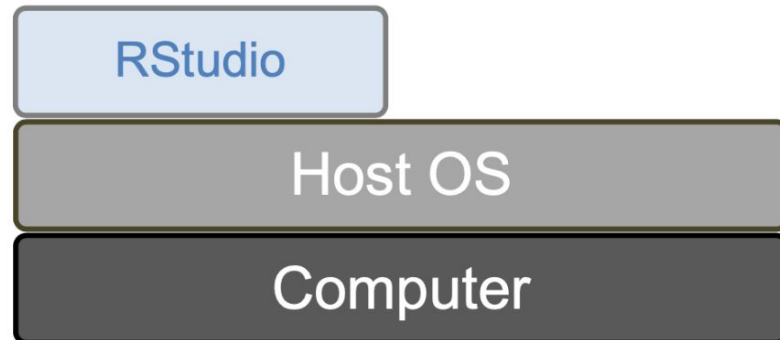
Unix-based





We started with a computer using a specific OS...





We started with a computer using a specific OS...

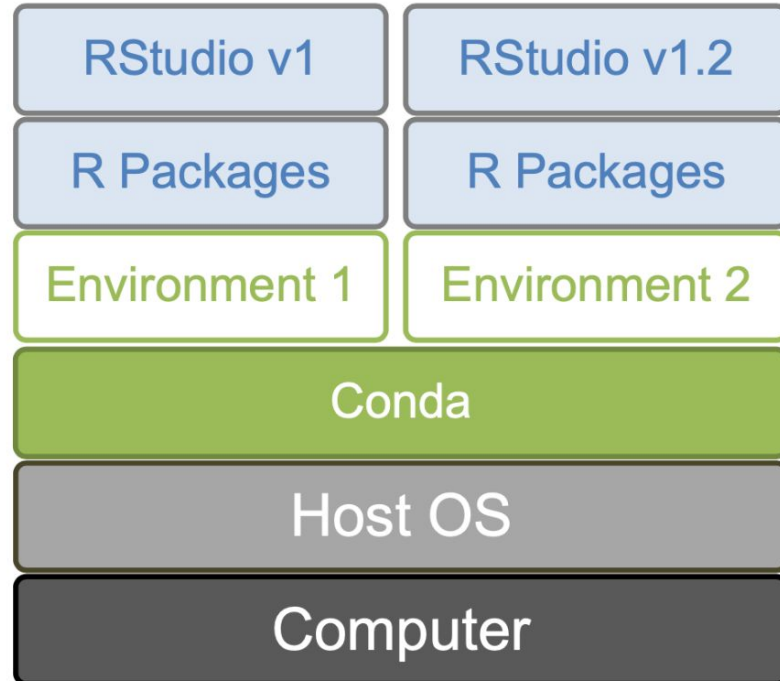
And inside this environment, we installed a new application.



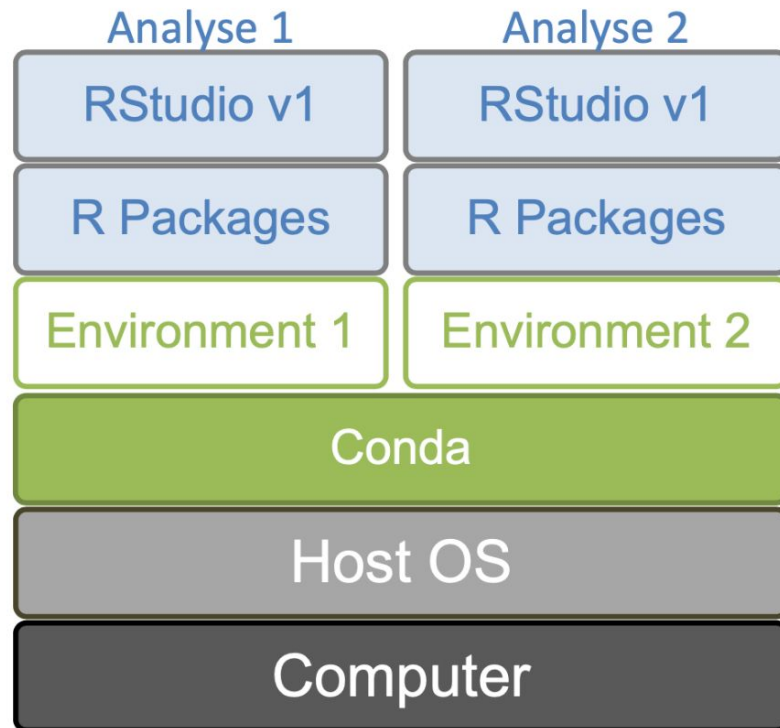
Usually dependencies of different applications don't interfere.
But what if we want to test the latest version of our favourite tool?
There might be conflicts...



Usually dependencies of different applications don't interfere. But what if we want to test the latest version of our favourite tool? There might be conflicts...

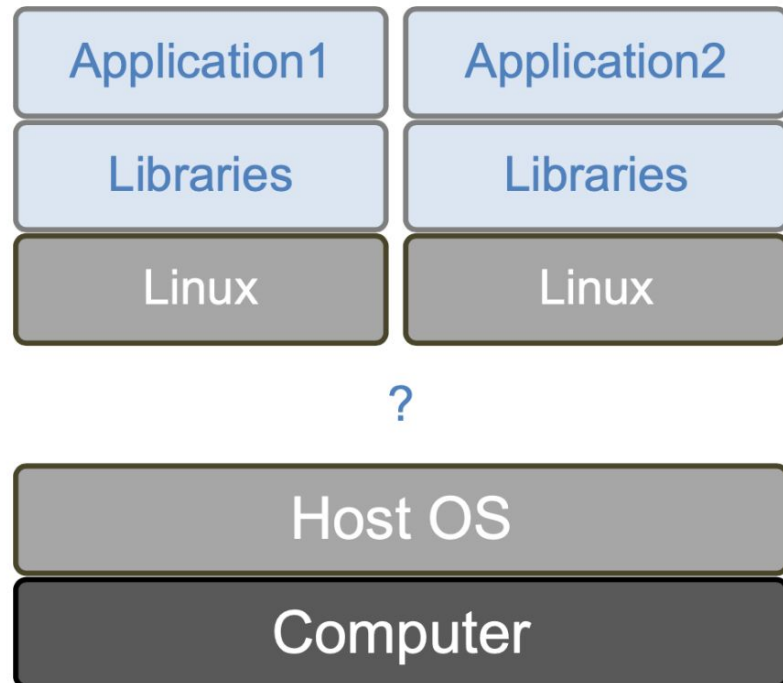


Idea : create separated environments for each application.

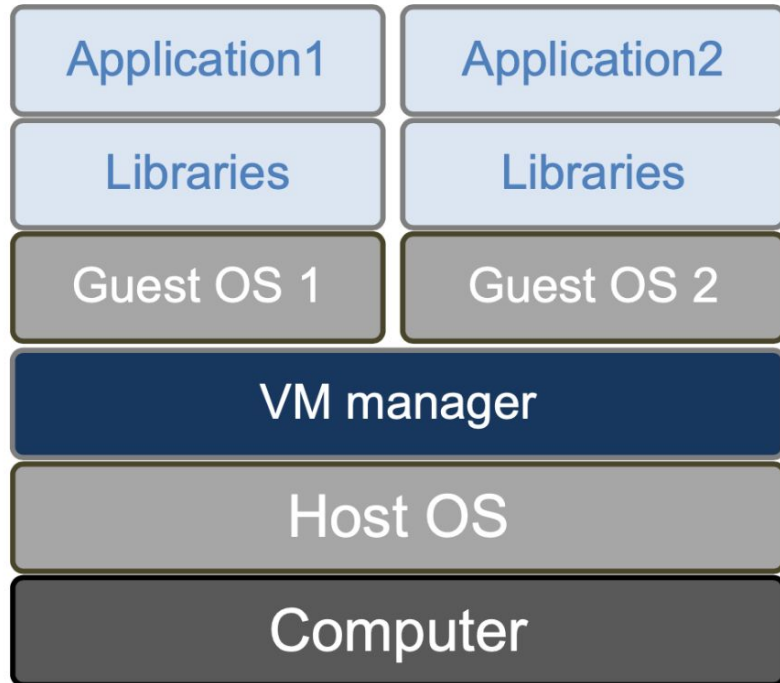


Idea : create separated environments for each application.

More versatile: create a new environment per analysis.



But what if we want to install a software from a different OS?

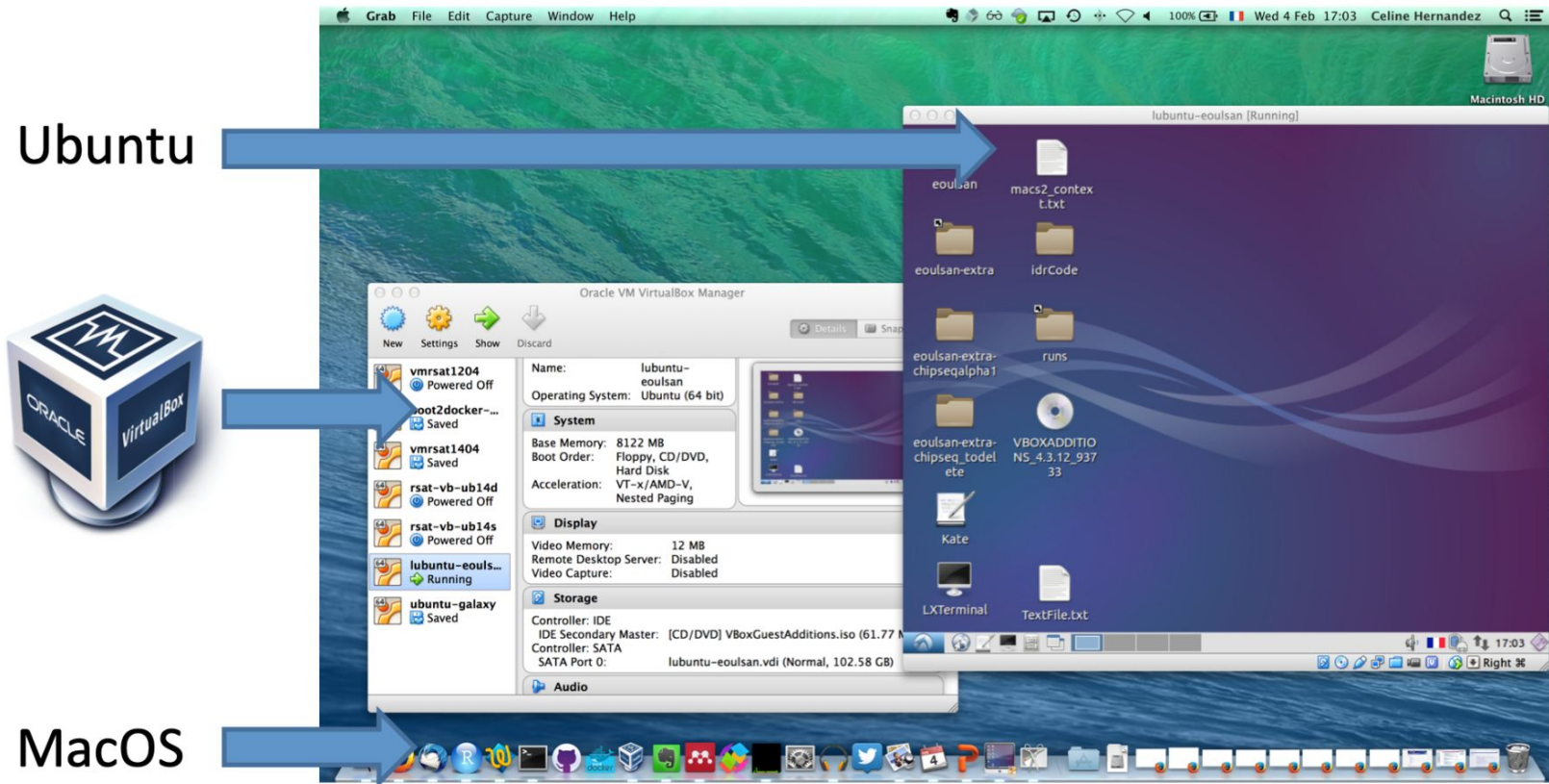


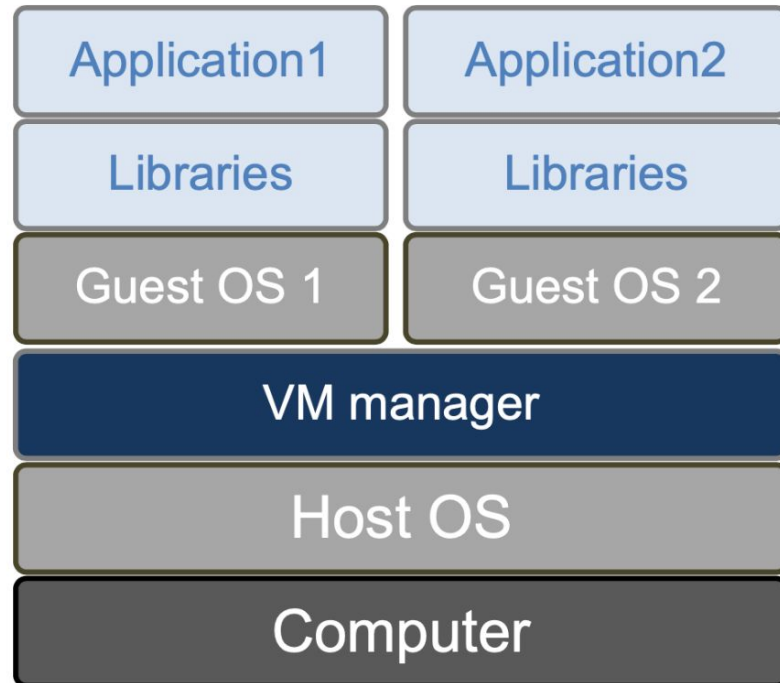
Idea: use virtual machines

Pros:

- Each application gets a completely different and independent environment
- Virtual machines can be transferred to another computer (using the same manager)

Encapsulation : hardware virtualization



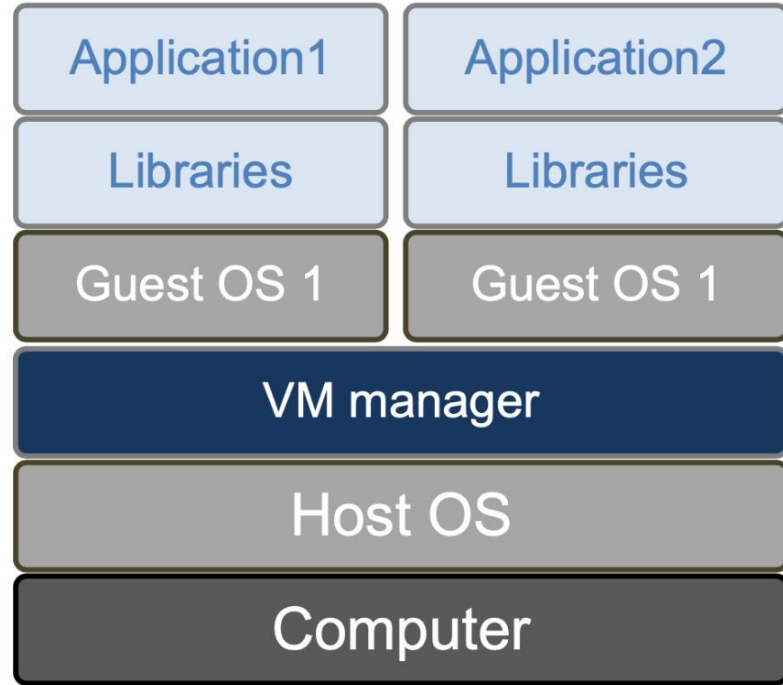


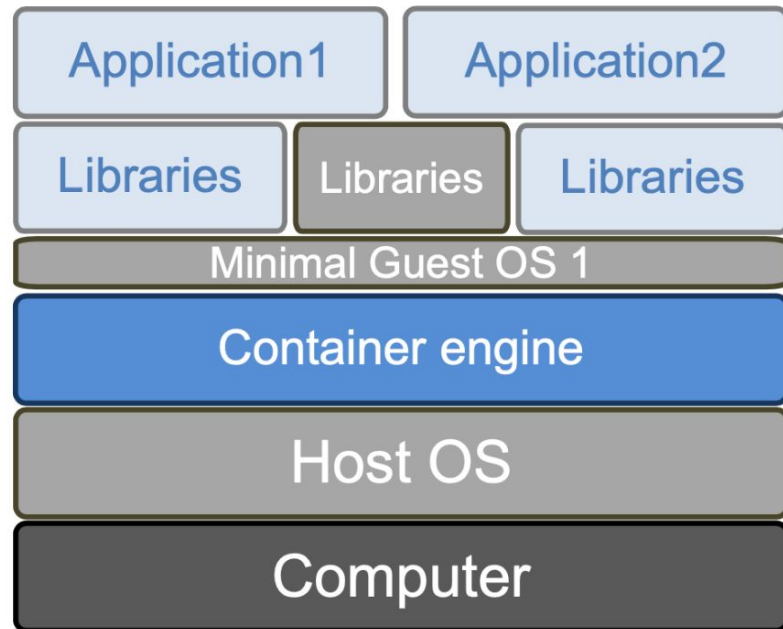
Idea: use virtual machines

Pros: transferable independent environments

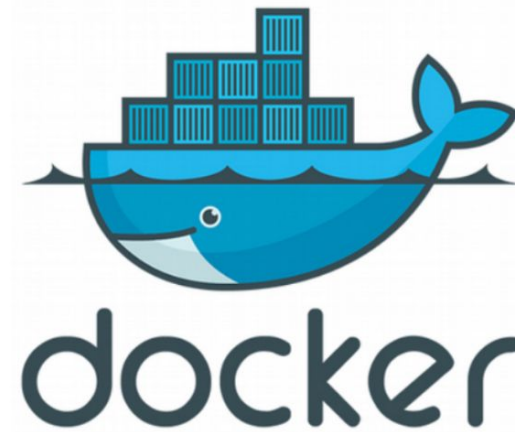
Cons:

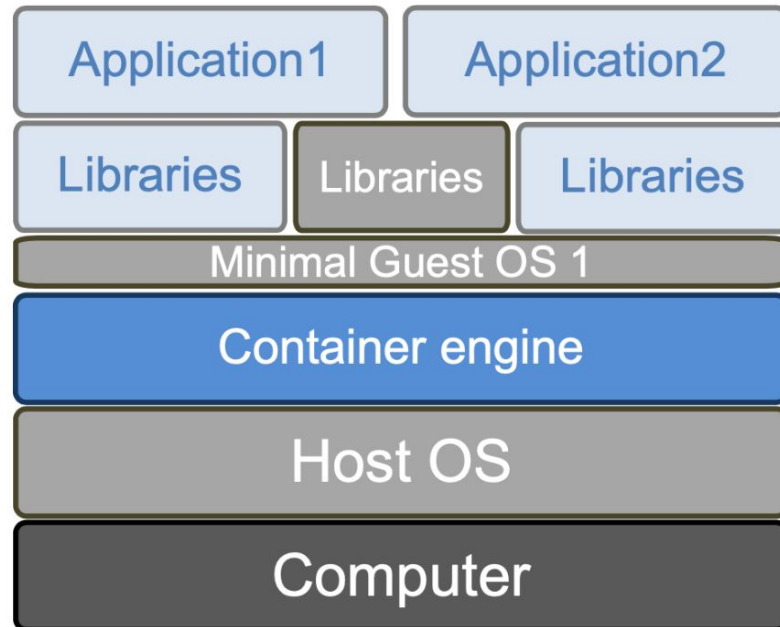
- Redundancy between VMs
- Heavy to set up
- No automation





Idea: "trick" applications into believing that they are in a different OS than the host's
Avoid redundancy.

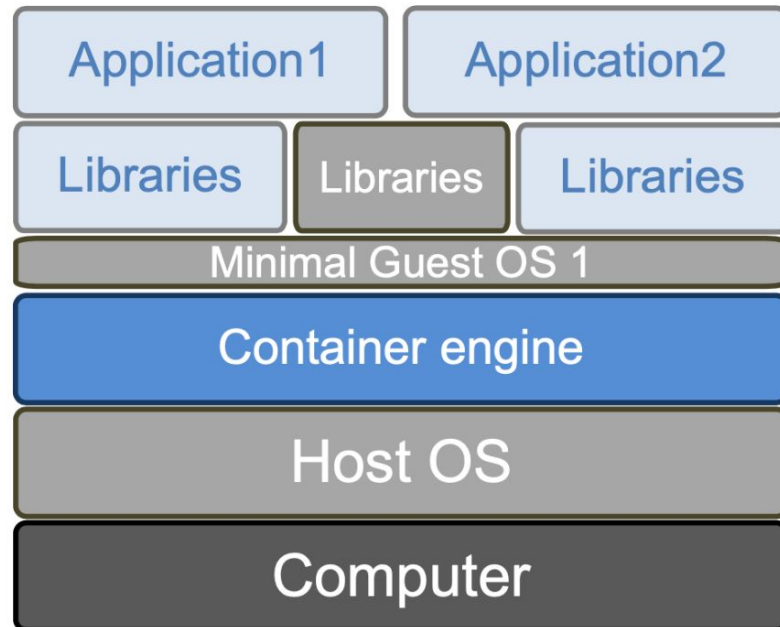




OS virtualisation vs hardware virtualisation

Pros:

- Speed
 - ▶ Installation is faster
 - ▶ No boot time
- Lightweight
 - ▶ Minimal base OS
 - ▶ Minimal libraries and application set
- Easy sharing of applications



Cons:

- Singularity to use images on a cluster
- Changes of policies of the Docker company



Update of the Docker Image retention policy (13/08/2020)

What is a container image retention limit and how does it affect my account?

Image retention is based on the activity of each individual image stored within a user account. If an image has not either been pulled or pushed in the amount of time specified in your subscription plan, the image will be tagged “inactive.” Any images that are tagged as “inactive” will be scheduled for deletion. Only accounts that are on the **Free** individual or organization plans will be subject to image retention limits. A new dashboard will also be available in Docker Hub that offers the ability to view the status of all of your container images.

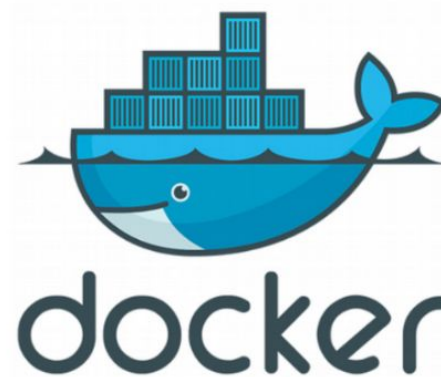
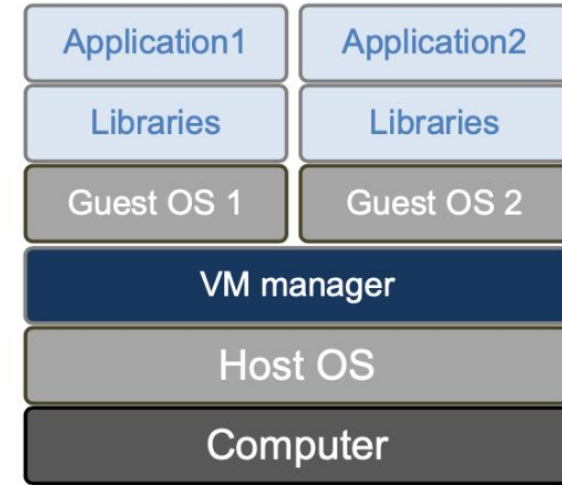
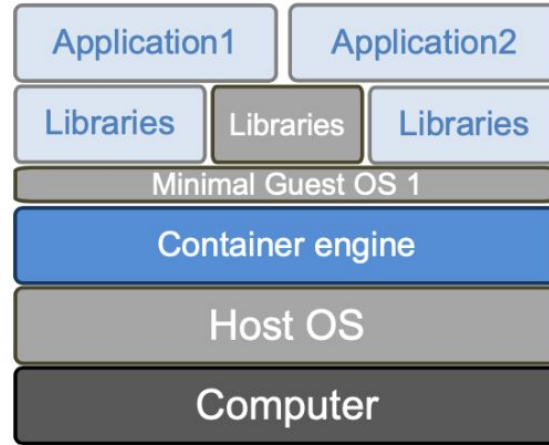
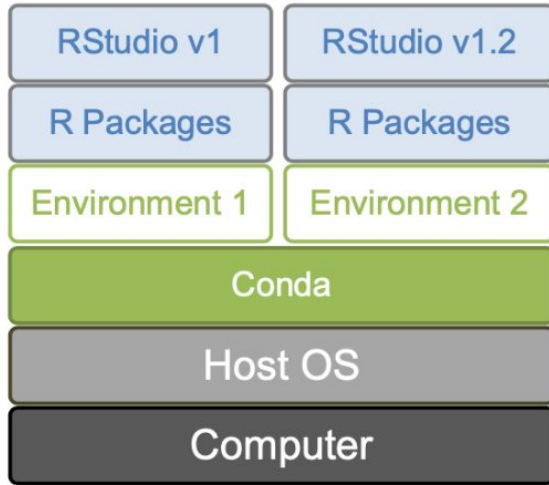
What are the new container image retention limits?

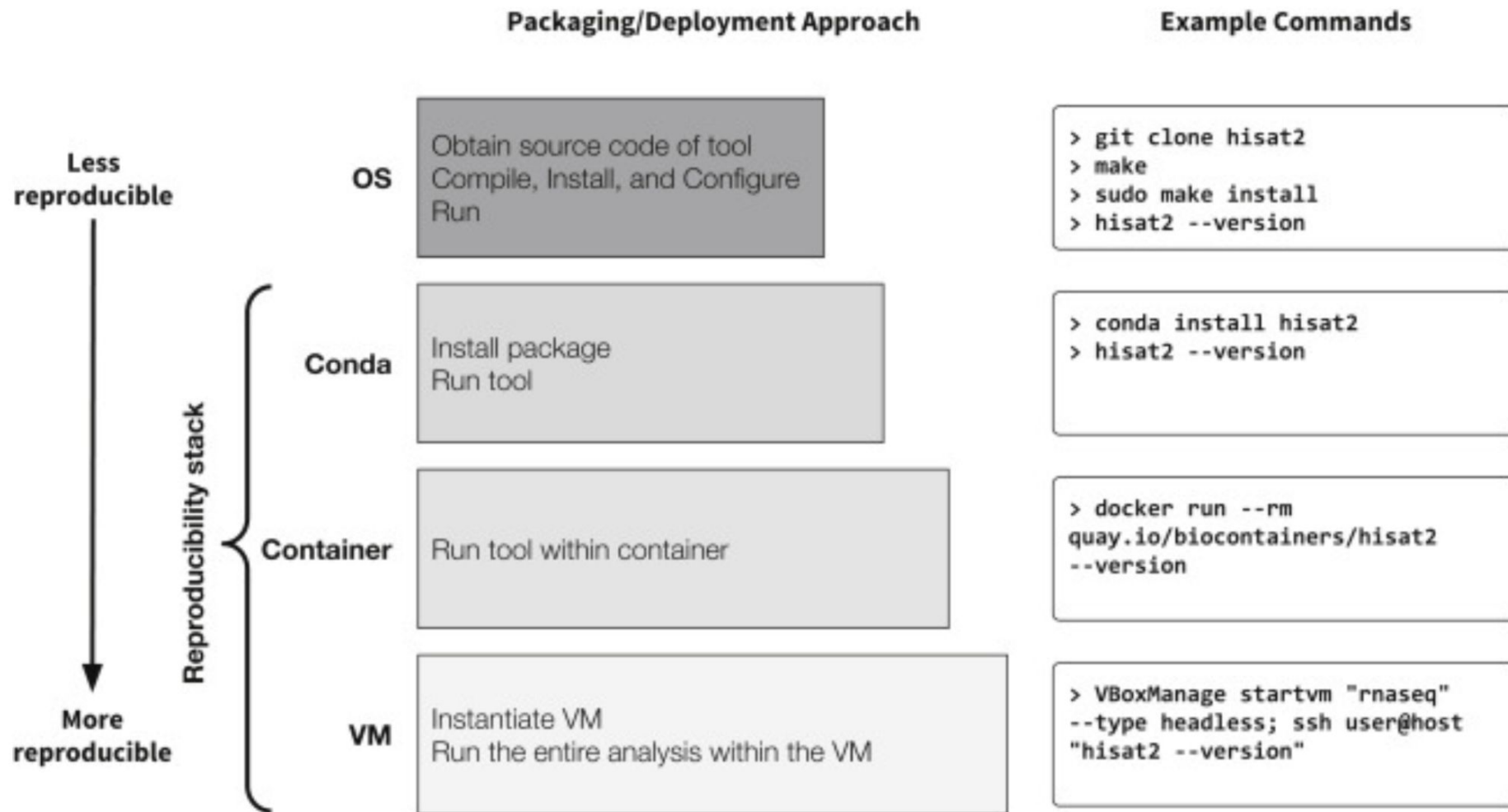
Docker is introducing a container image retention policy which will be enforced starting November 1, 2020. The container image retention policy will apply to the following plans:

- Free plans will have a 6 month image retention limit
- Pro and Team plans will have unlimited image retention

<https://www.docker.com/pricing/retentionfaq>

Encapsulation

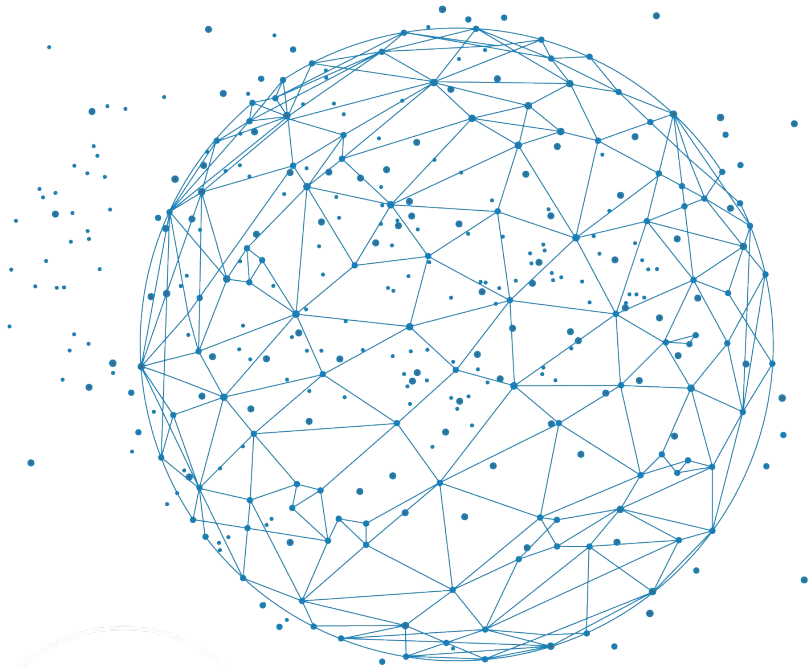




Practical Computational Reproducibility in the Life Sciences - Björn Grüning et al (2018)



Docker



INSTITUT FRANÇAIS DE BIOINFORMATIQUE





Docker is not very “old”

- First commit January 2013
- First version March 2013
- Version 1.0 in June 2014

But its adoption was fast

- Officially packaged in Ubuntu since 2014 (v14.04)



Image



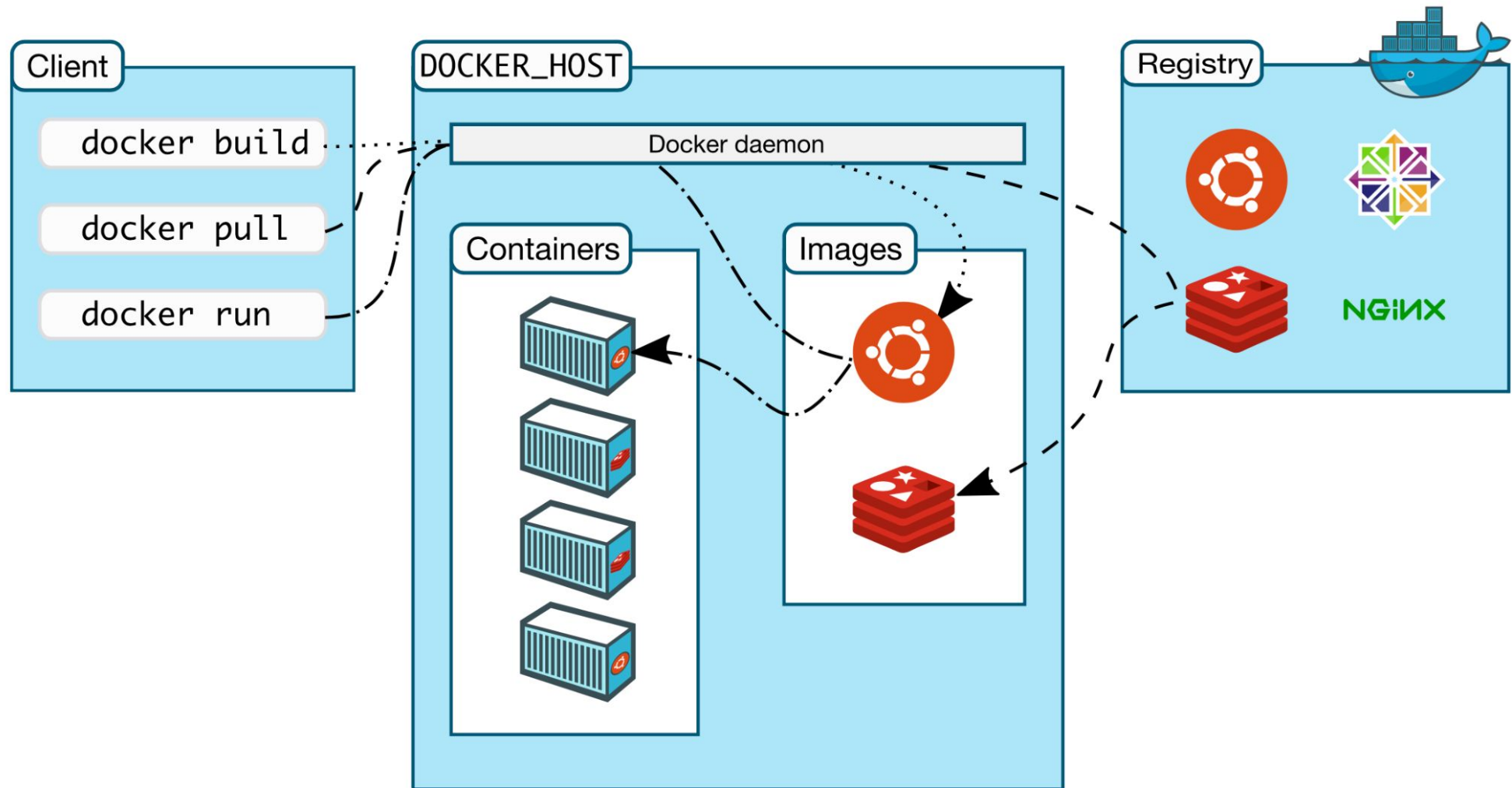
- Set of libraries and functions
- Fixed. Cannot be modified
- Can be stored/shared online
- Can be automatically built

Container



- "Active image"
- Can be modified (interactive)
- Can be turned into an image
- One image, many containers

What is Docker?



(<https://docs.docker.com/get-started/overview/>)






What is Docker?



Docker Store is the new place to discover public Docker content. [Check it out](#) →

Explore Help [Sign up](#) [Sign in](#)

Explore Official Repositories

| | | | |
|---|---------------|---------------|---------------------------------|
|  nginx official | 5.3K STARS | 10M+ PULLS | > DETAILS |
|  redis official | 3.4K STARS | 10M+ PULLS | > DETAILS |
|  busybox official | 924 STARS | 10M+ PULLS | > DETAILS |
|  ubuntu official | 5.5K STARS | 10M+ PULLS | > DETAILS |
|  registry official | 1.3K STARS | 10M+ PULLS | > DETAILS |

(<https://hub.docker.com/>)



Usermade images (1/2)

The screenshot shows the Docker Hub profile for 'genomicpariscentre'. The profile includes a bio: 'Genomic Paris Centre', location 'Paris', website 'http://genomique.biologie.ens.fr/', and a join date of 'June 2014'. Below the bio is a table of Docker images:

| Repository Name | Stars | Pulls | Details |
|--|-------|-------|-----------|
| genomicpariscentre/star public automated build | 1 | 1.2K | > DETAILS |
| genomicpariscentre/bcl2fastq2 public automated build | 0 | 1.2K | > DETAILS |
| genomicpariscentre/blast2 public automated build | 0 | 765 | > DETAILS |
| genomicpariscentre/bcbio-nextgen public automated build | 0 | 451 | > DETAILS |
| genomicpariscentre/fastqc public automated build | 0 | 404 | > DETAILS |
| genomicpariscentre/bowtie2 public automated build | 0 | 308 | > DETAILS |
| genomicpariscentre/samtools public automated build | 0 | 304 | > DETAILS |
| genomicpariscentre/eoulsan public automated build | 2 | 231 | > DETAILS |

(<https://hub.docker.com/u/genomicpariscentre/>)



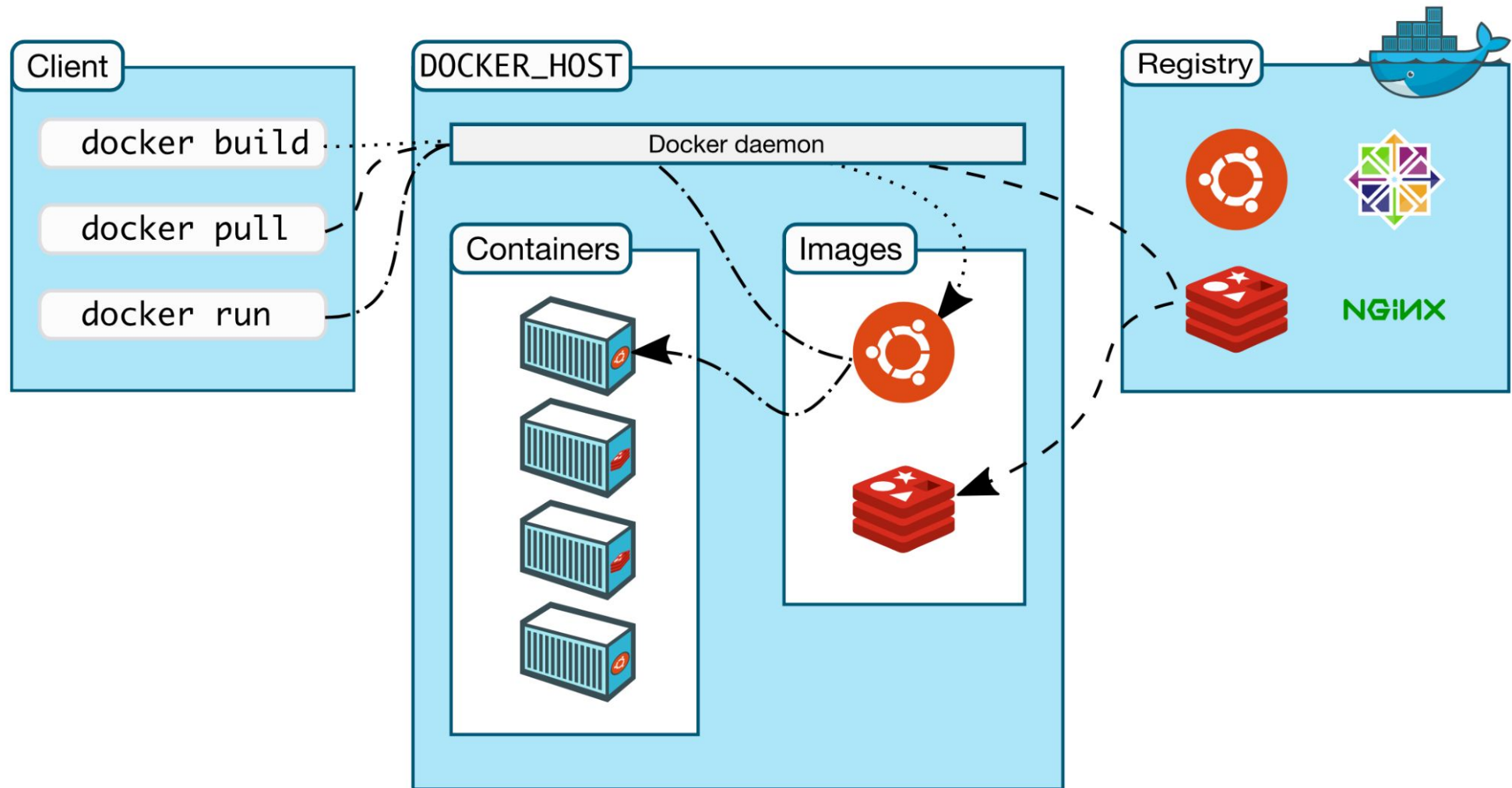
Usermade images (2/2)

Be critical!

The screenshot shows the Docker Hub interface for the repository `genomicpariscentre/samtools`. The page is titled "PUBLIC | AUTOMATED BUILD" and shows the repository name with a star icon and "Last pushed: 2 years ago". The navigation tabs include "Repo Info", "Tags", "Dockerfile", and "Build Details". The main content area is divided into two columns. The left column contains a "Short Description" and a "Full Description", both stating: "Samtools is a processor of sequence alignments for SAM and BAM formats". The right column contains a "Docker Pull Command" section with the command `docker pull genomicpariscentre/samtools`, an "Owner" section with the profile picture and name "genomicpariscentre", and a "Source Repository" section with a link to `GenomicParisCentre/dockerfiles`.

(<https://hub.docker.com/r/genomicpariscentre/samtools/>)

What is Docker?



(<https://docs.docker.com/get-started/overview/>)



Other commands :

- `docker images` : list images available locally
- `docker ps` : status of containers
- `docker rm` : delete a container
- `docker rmi` : delete an image
- ...

(More details during the practical session.)

