Orchestrating applications with TOSCA and Docker

Luca Rinaldi

University of Pisa

June 2017

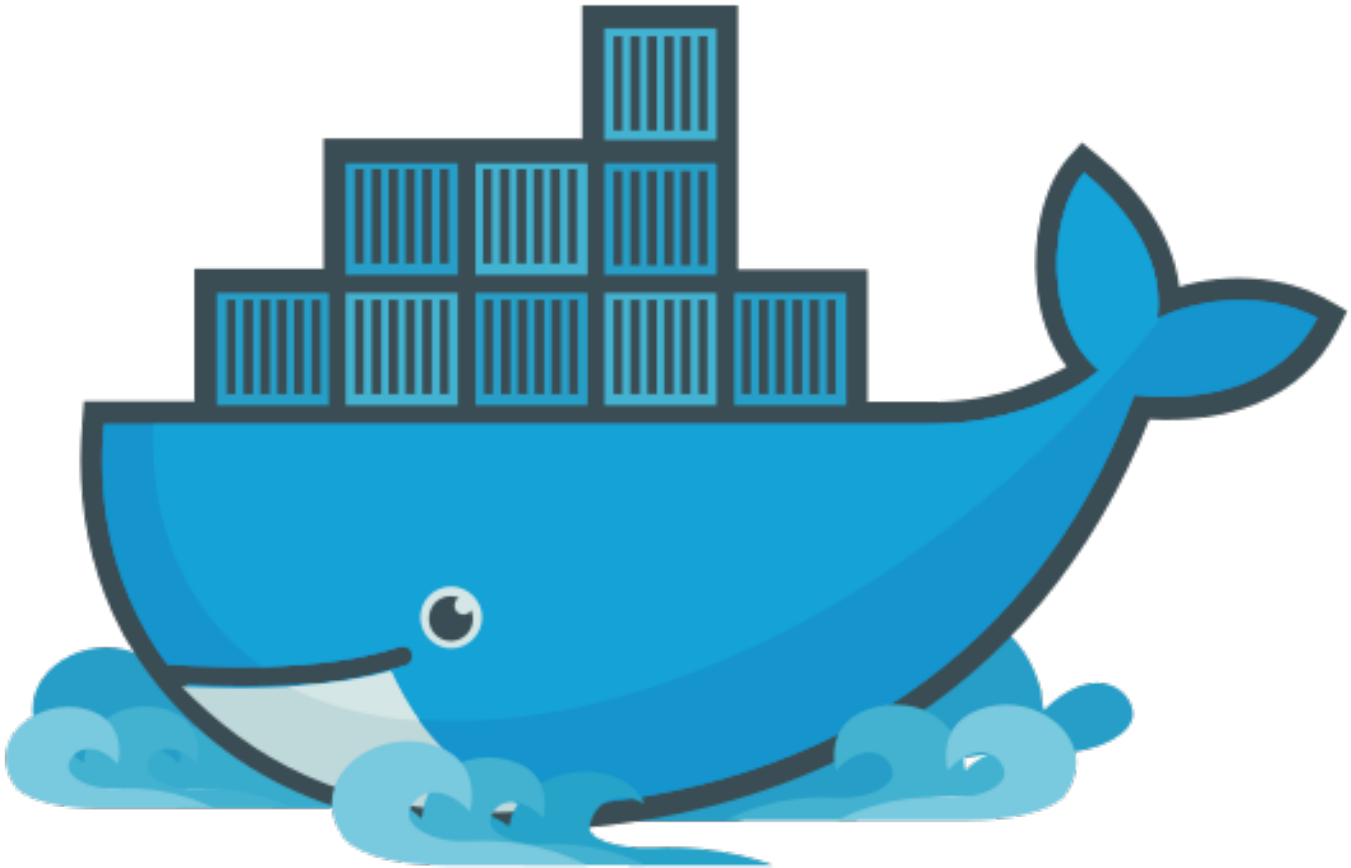
Table of Contents

Software deployment

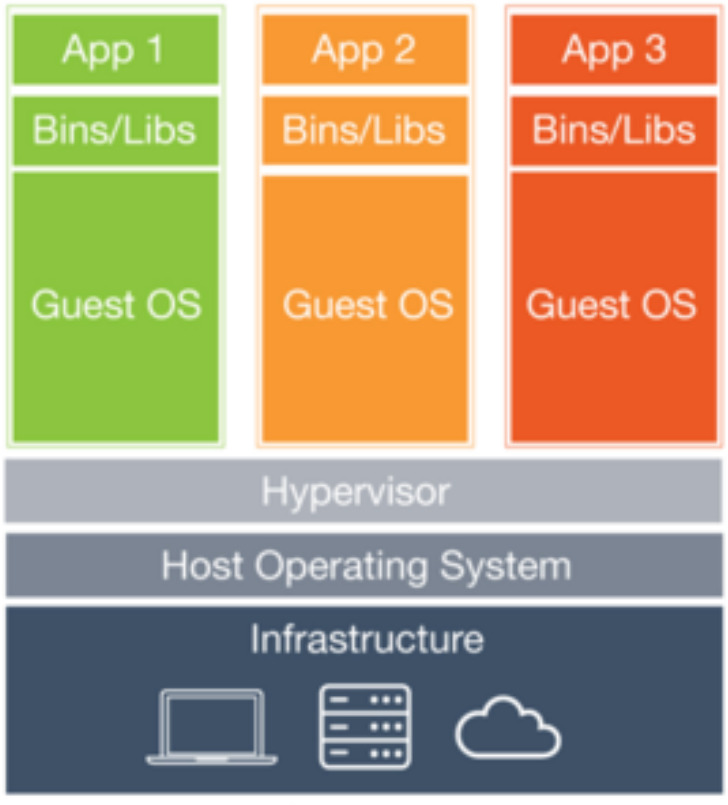
The execution of all the activities that make a software system available to use.
Nowadays strictly related to the cloud infrastructure.
Need of a way to express all the **requirements** that the application needs to run.

Table of Contents

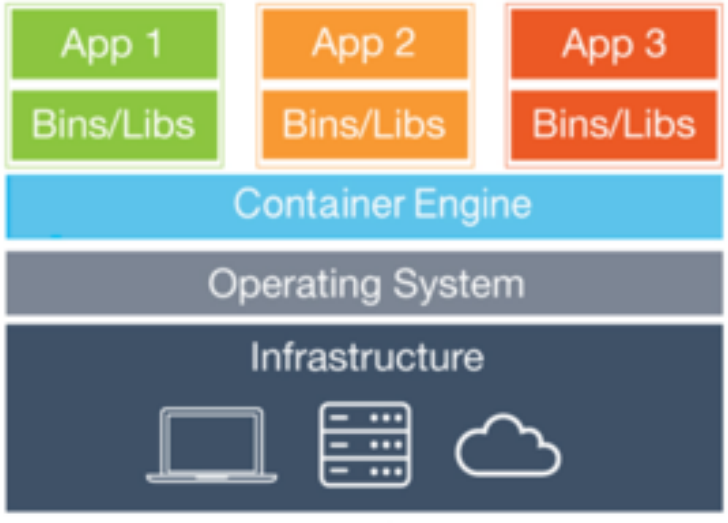
Docker



Docker: What?

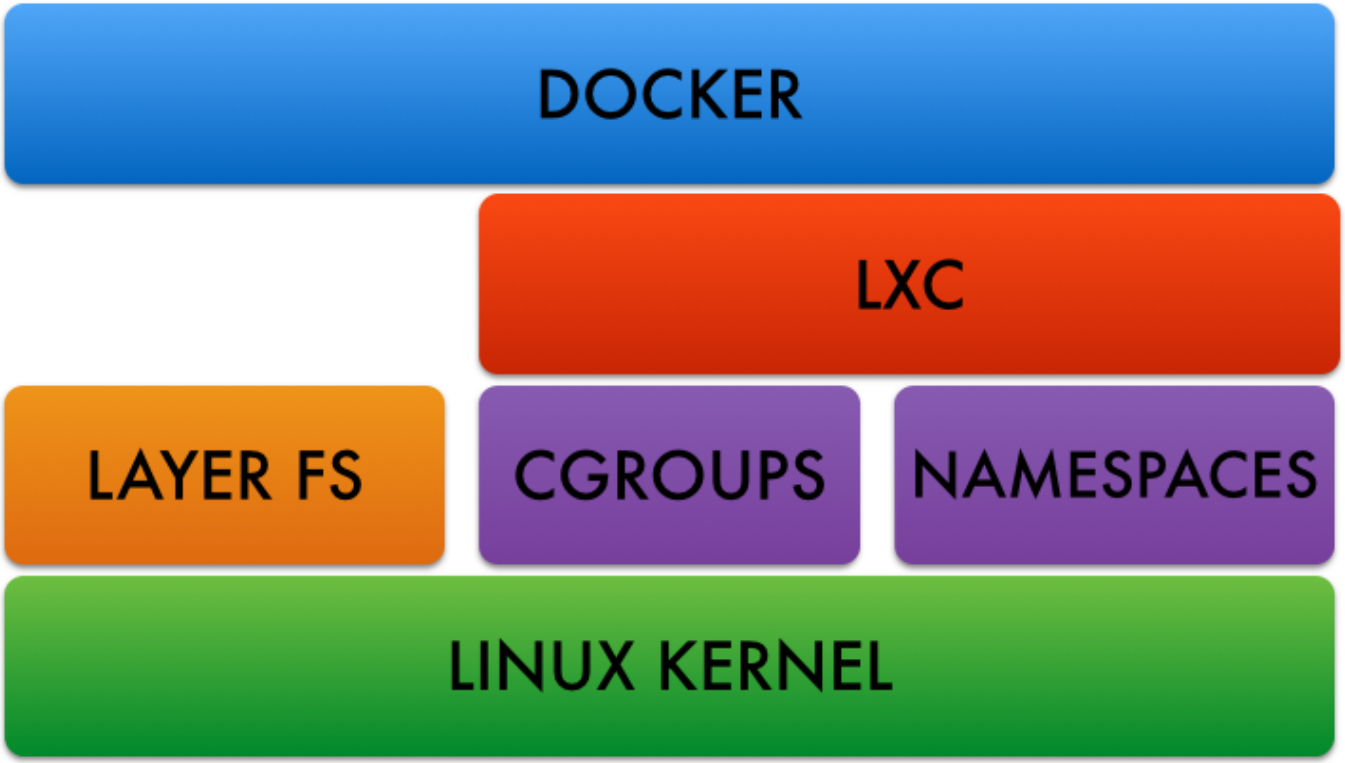


Hypervisor-based Virtualization



Container virtualization

Docker: Architecture



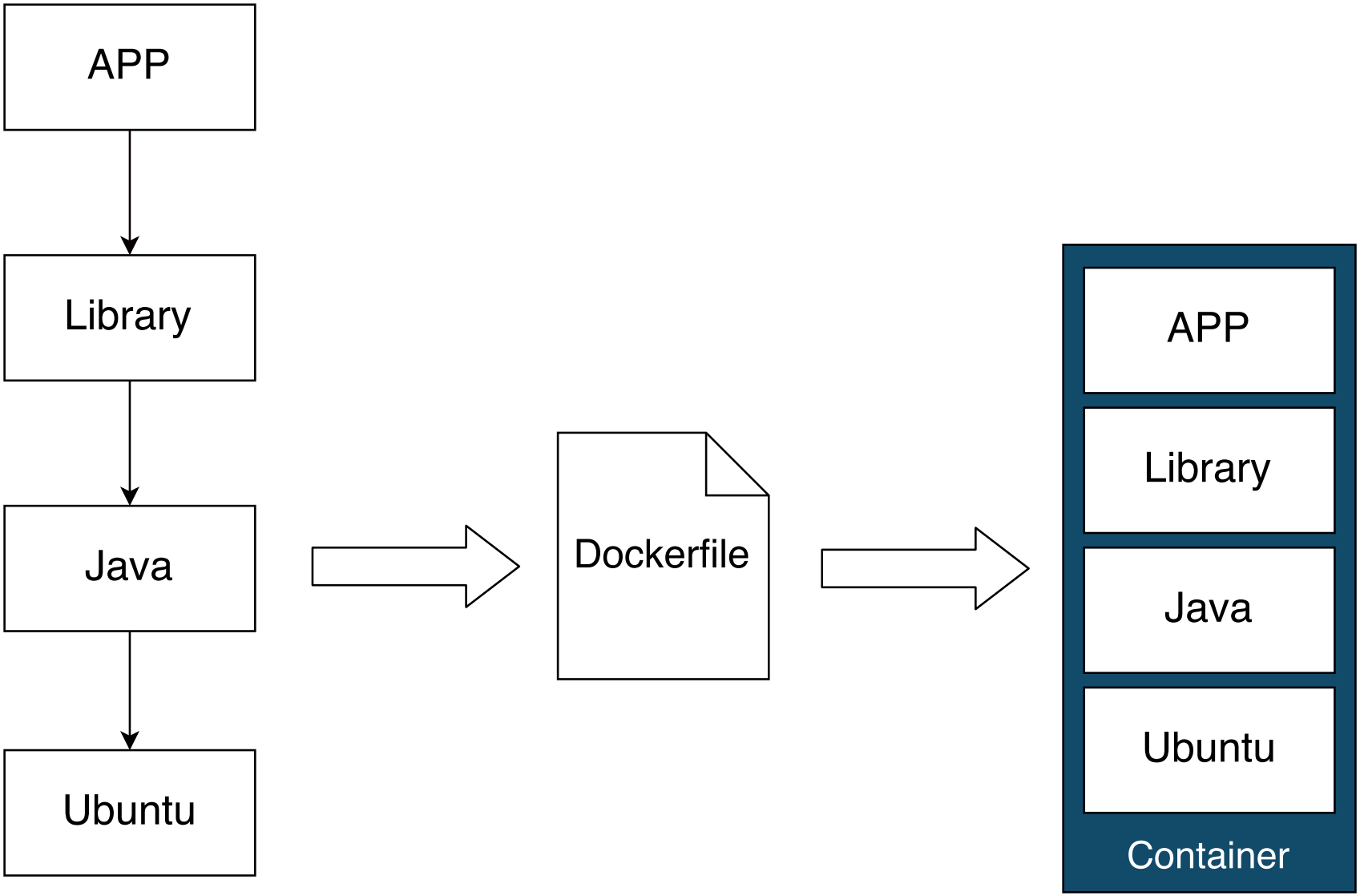
- LXC
- LAYER FS
- CGROUPS
- NAME SPACE
- LINUX KERNEL

Docker: main comcepts

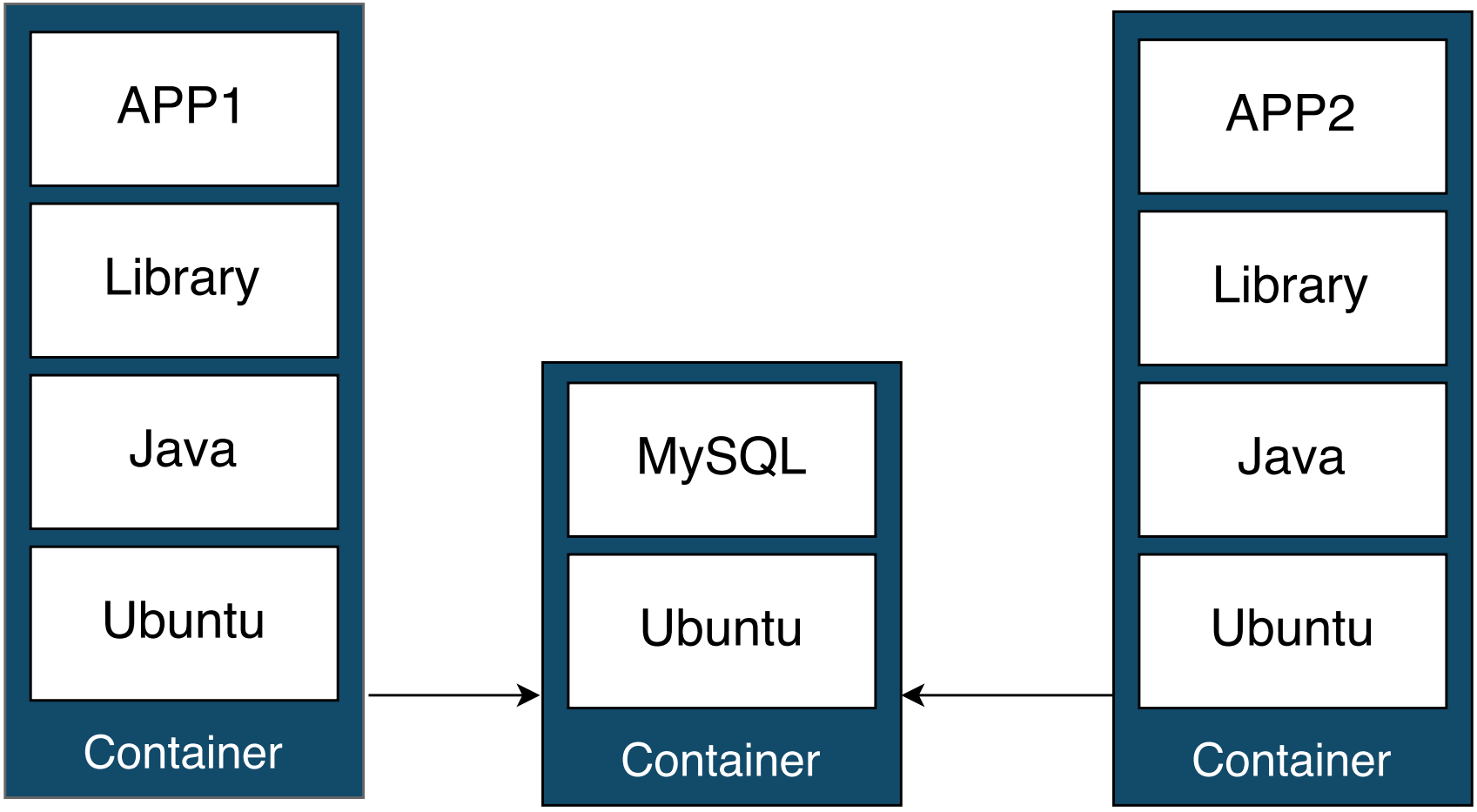
Main concept of the Docker platform

- **Dockerfile**, a scripts to generate an Image
- **Docker Image**, a LAYER FS archive whit all the data
- **Docker Container**, Running instance of a Docker Image
- **Docker Volume**, a persistent data storage
- **Docker Hub**, a Database of Docker Image open to comunity

Docker: for deploy application



Docker: multicontainer



Problems

- Poor application orchestration
- The container hide the components that it contains
- Can orchestrate only container

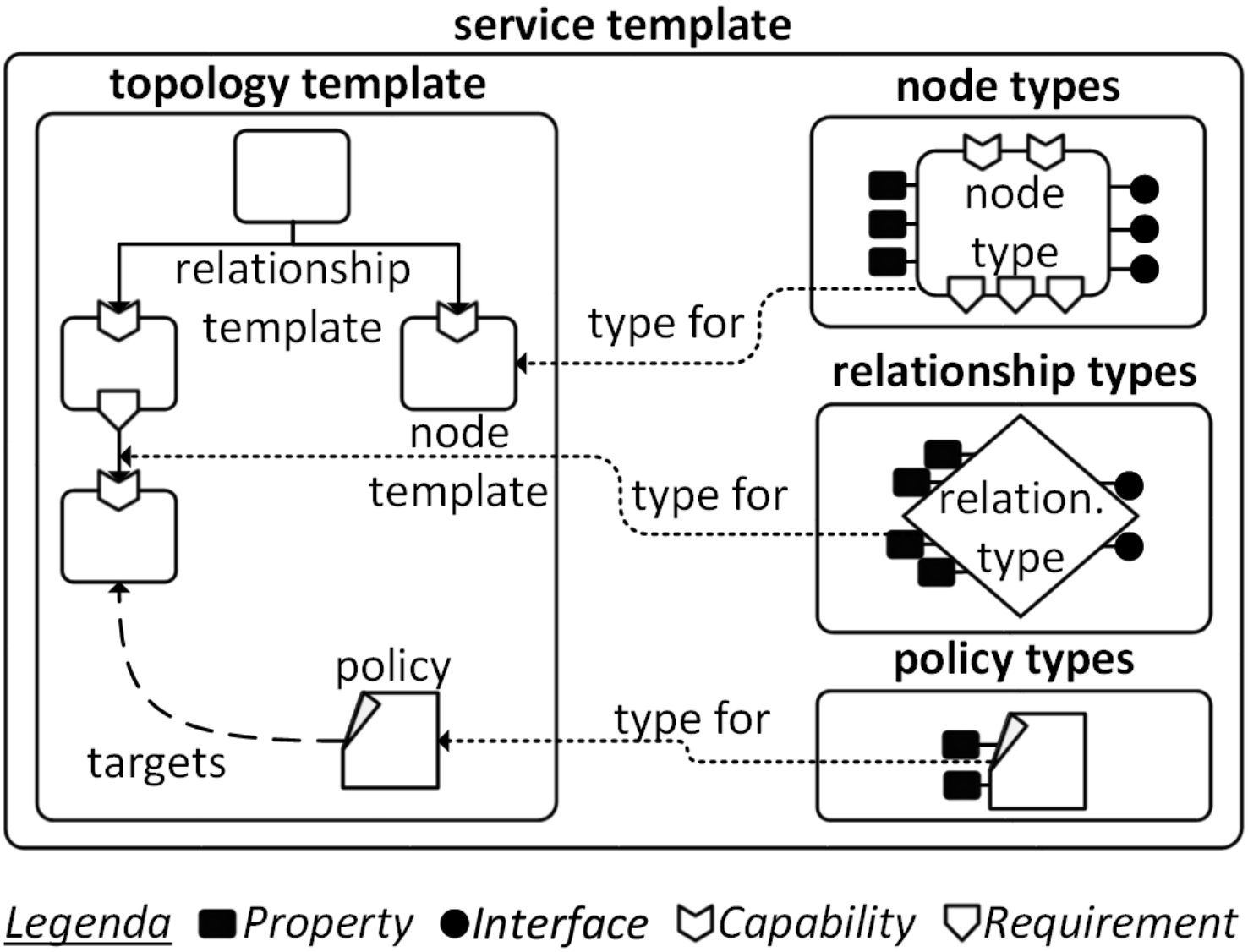
Table of Contents

TOSCA: What?

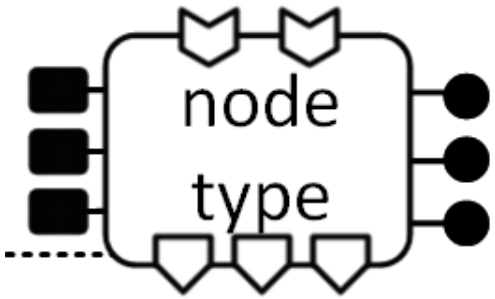
OASIS standard language to describe the topology of an application, with its components and relationships.

Describe every part of your application!

TOSCA: main concept



TOSCA: Node type



- **requirements**, what the node require
- **capabilities**, what the node offer
- **properties**, the properties of the node
- **interfaces**, the operation to implement
- **artifacts**, the data need to use the node

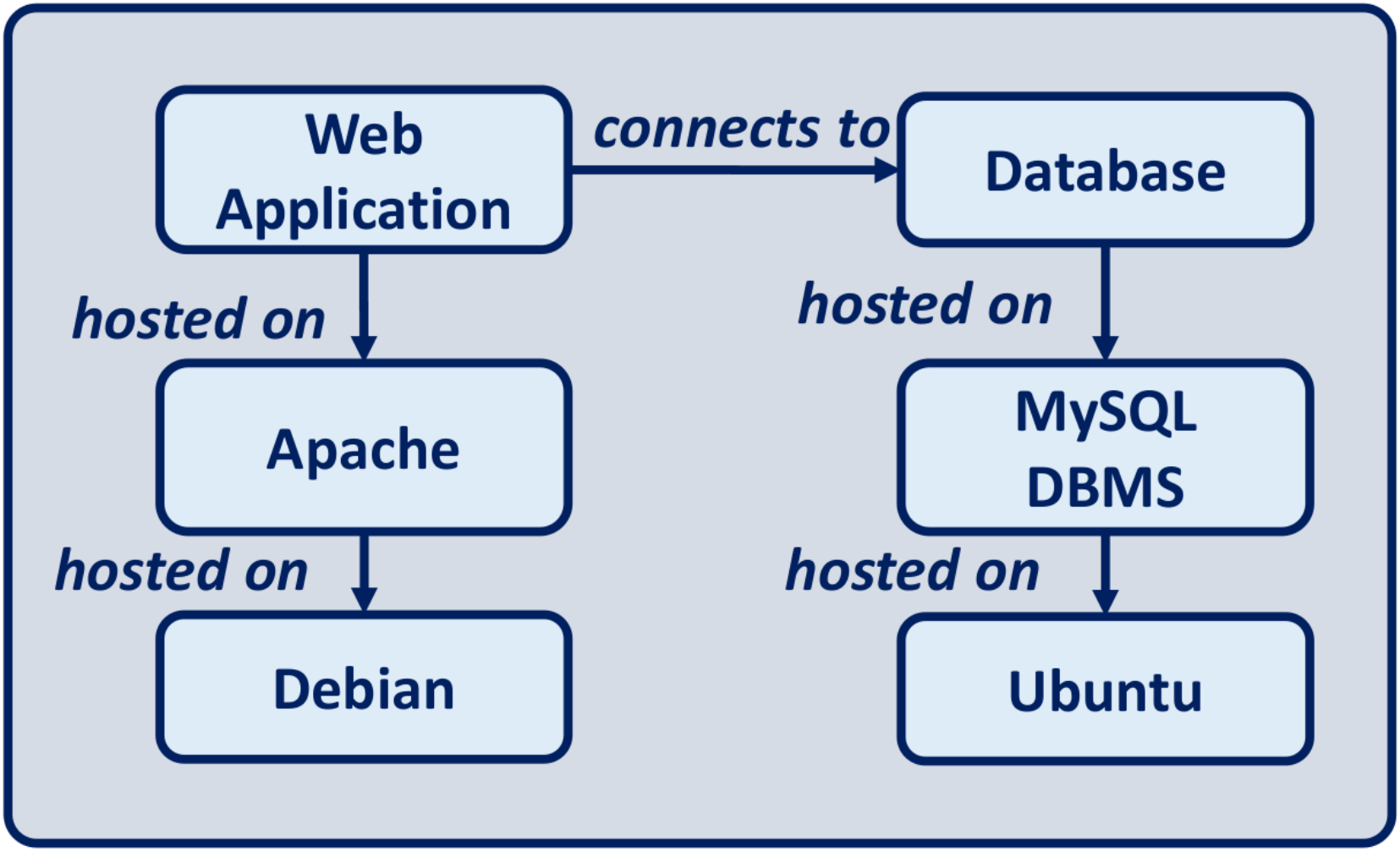
TOSCA main concepst

- the description use a YAML destription
- CSAR its an archive with the YAML description and all the artifacts

TOSCA: how it works

- create an archive (.CSAR) whit the description and artifacts
- send to acompatible infrastructure
- following the description and using the artifacts the application can deploy

Orchestration



Problems

- The specification can become too verbose
- A good standard but not a product
- Lack of engines which accept TOSCA description

Table of Contents

TosKer

An orchestration engine capable of deploying, on top of Docker, applications described in TOSCA YAML.

TosKer: what?

TosKer inputs a TOSCA description of a multi-component application, where some components are Docker containers and Docker volumes, and automatically deploys and orchestrates it using the Docker engine.

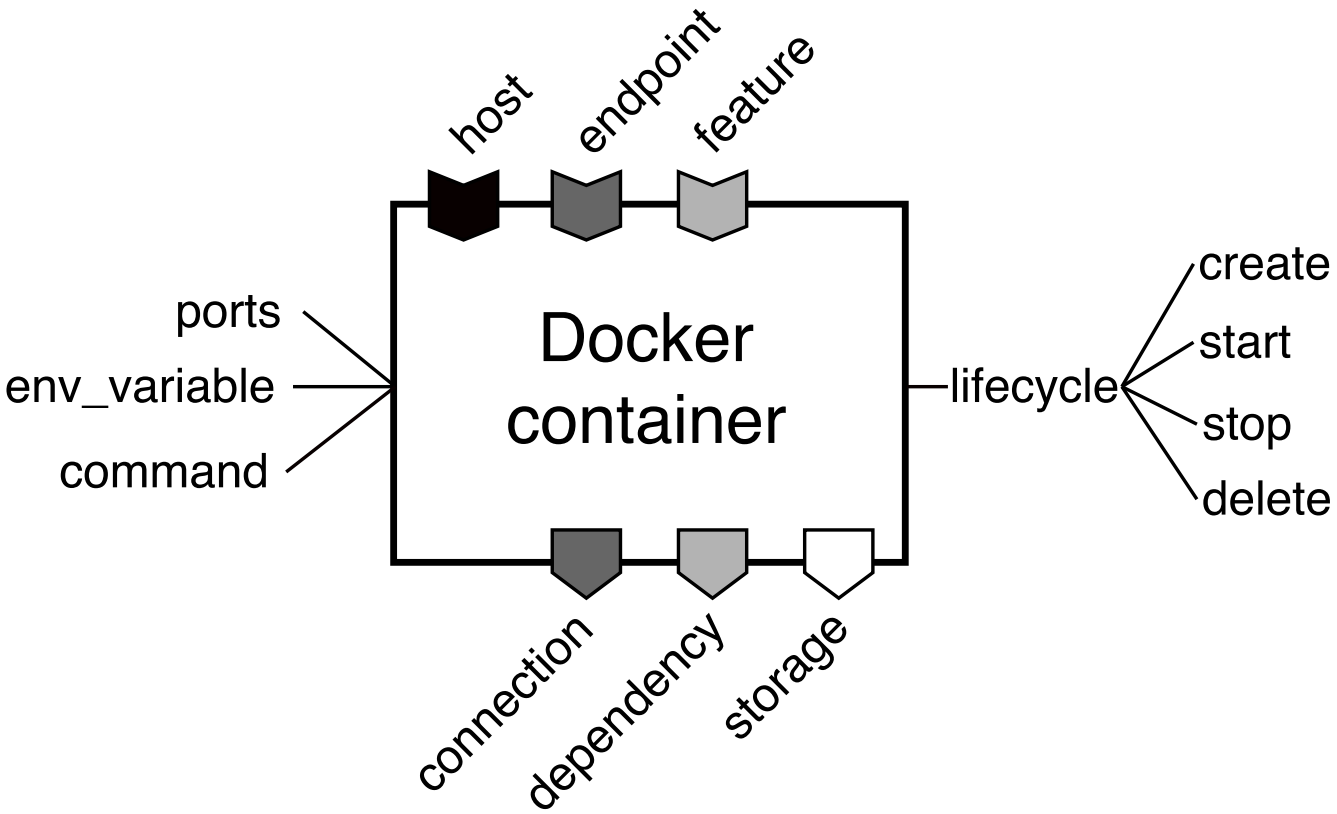
Describing applications in TosKer

- Applications are specified as a composition of the following components:
  - *Docker containers* `tosker.nodes.Container`
  - *Docker volumes* `tosker.nodes.Volume`
  - *Software* `tosker.nodes.Software`

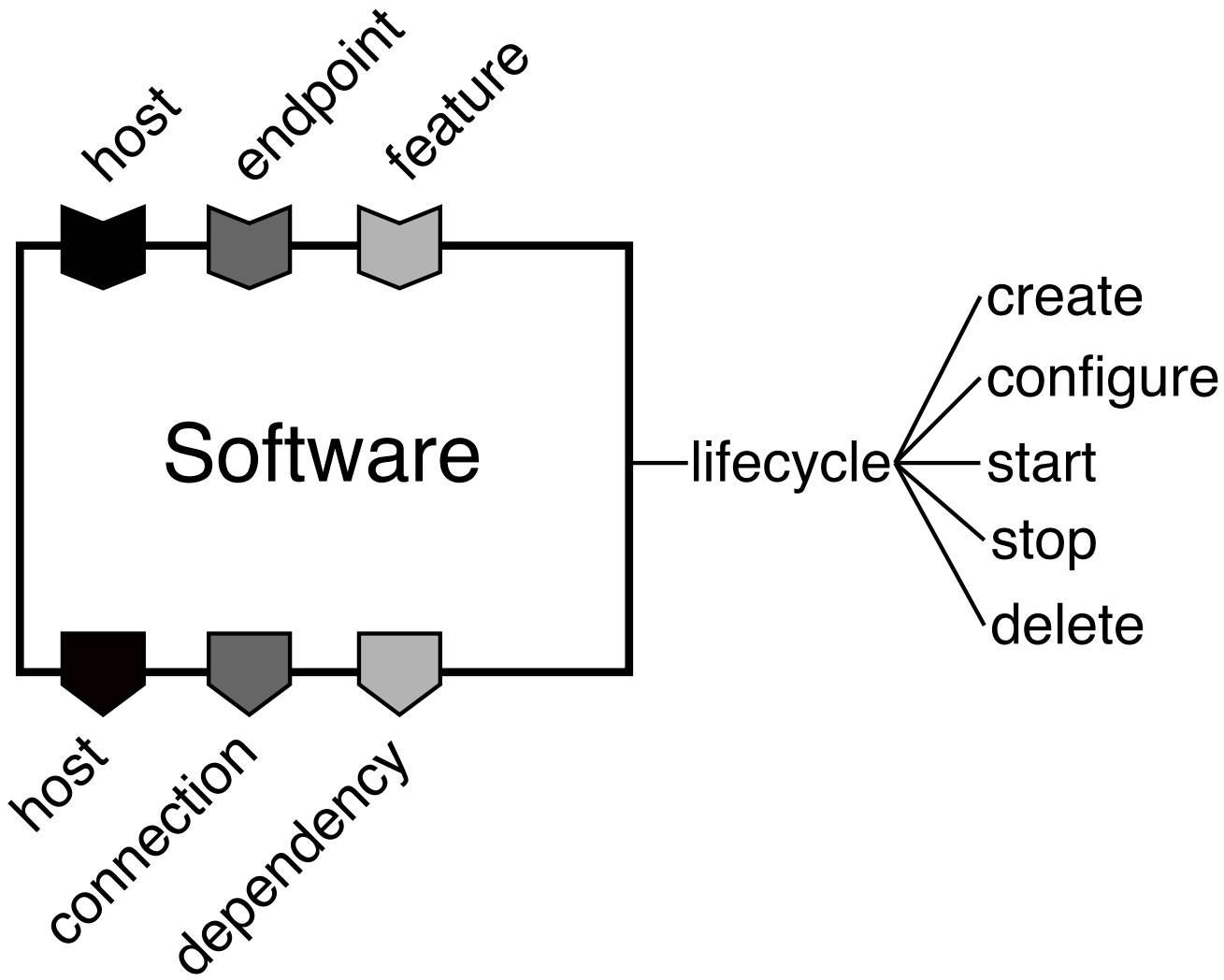
Describing applications in TosKer

- Applications are specified as a composition of the following components:
  - *Docker containers* `tosker.nodes.Container`
  - *Docker volumes* `tosker.nodes.Volume`
  - *Software* `tosker.nodes.Software`
- There can be the following relationships between components:
  - *hosted on* `tosca.relationships.HostedOn`
  - *connected to* `tosca.relationships.ConnectsTo`
  - *attached to* `tosca.relationships.AttachesTo`
  - *depending on* `tosca.relationships.DependsOn`

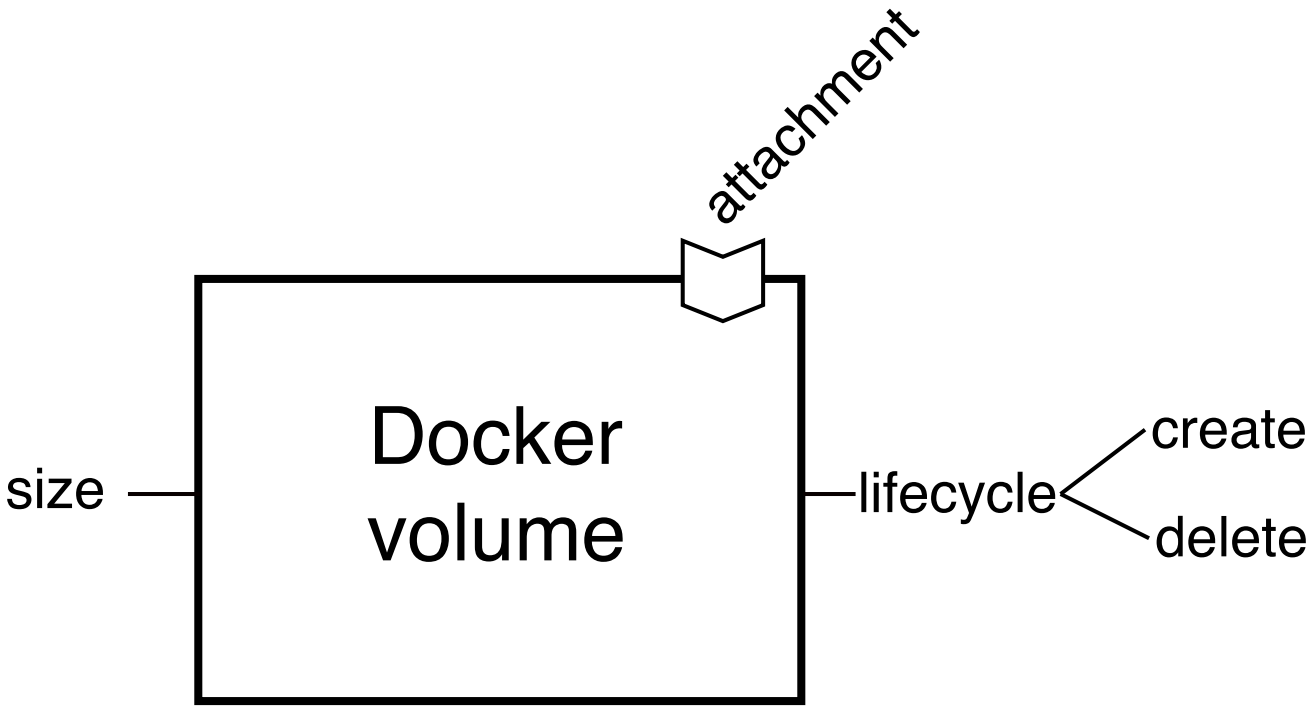
tosker.nodes.Container



tosker.nodes.Software



tosker.nodes.Volume

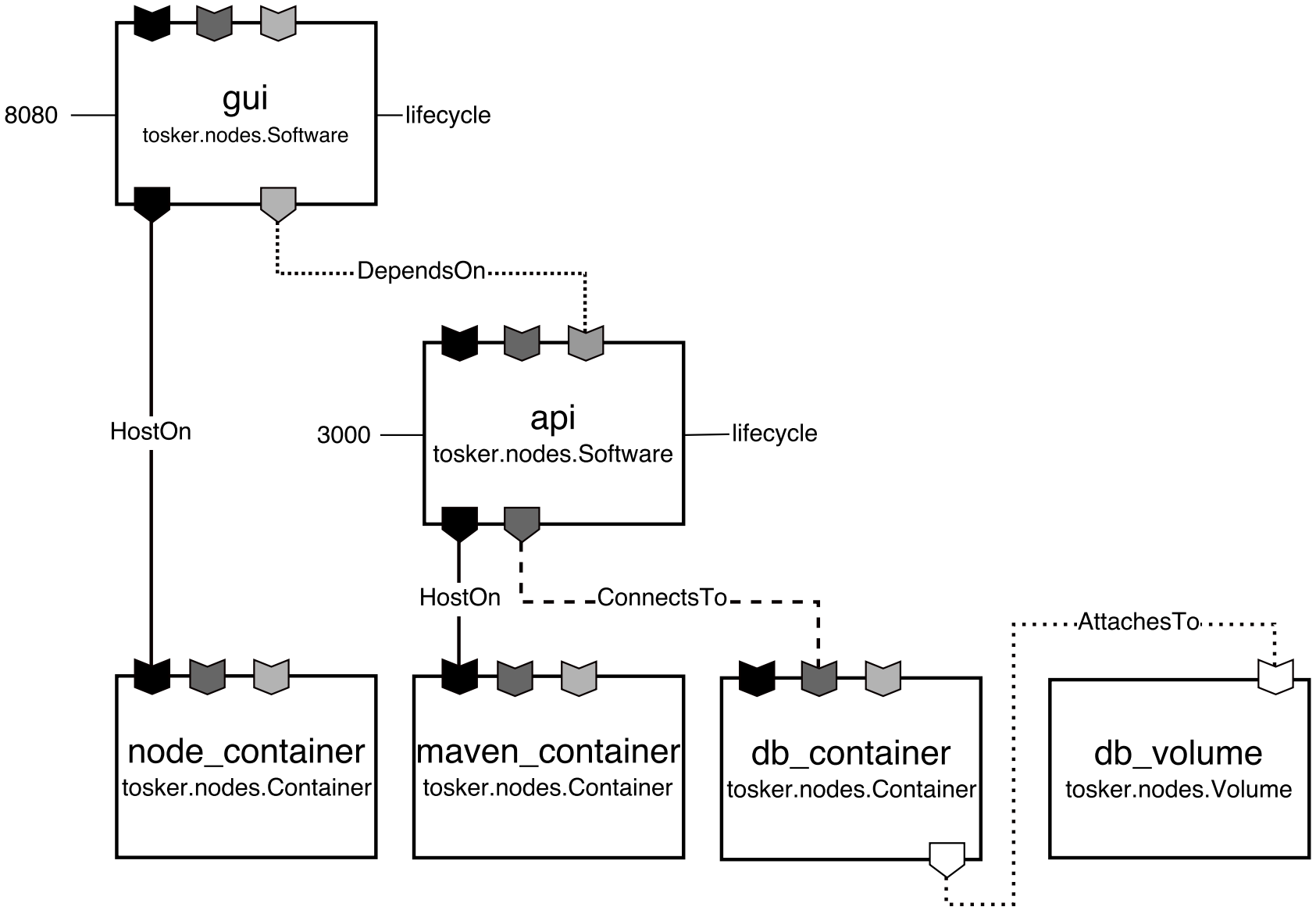


Constrains

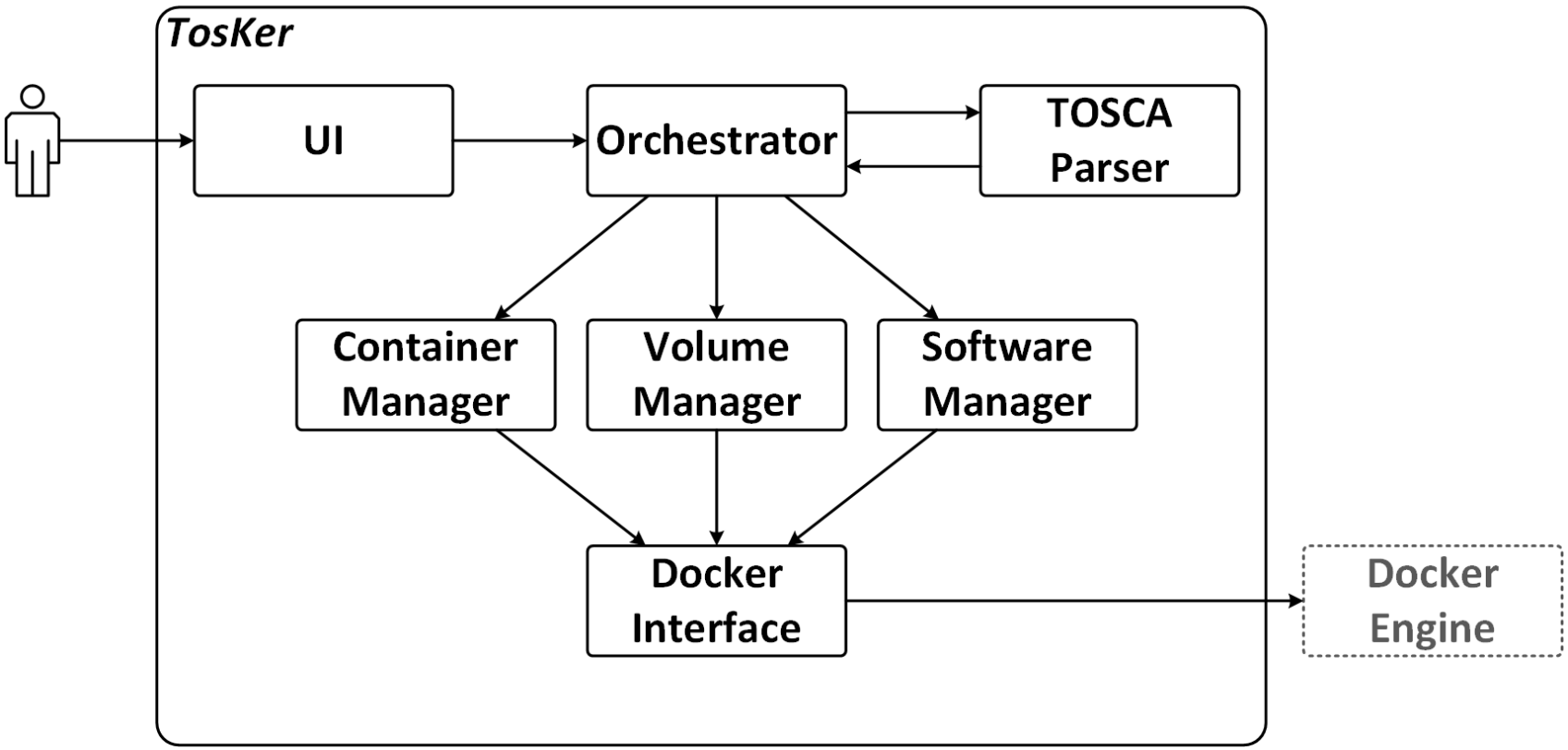
Each application must meet some constrains, e.g.,

- A *software* must be "*hosted on*" another *software* or a *Docker container*
- A *Docker container* and *Docker volume* cannot be "*hosted on*" other components
- Only *Docker containers* can be "*attached to*" *Docker volumes*

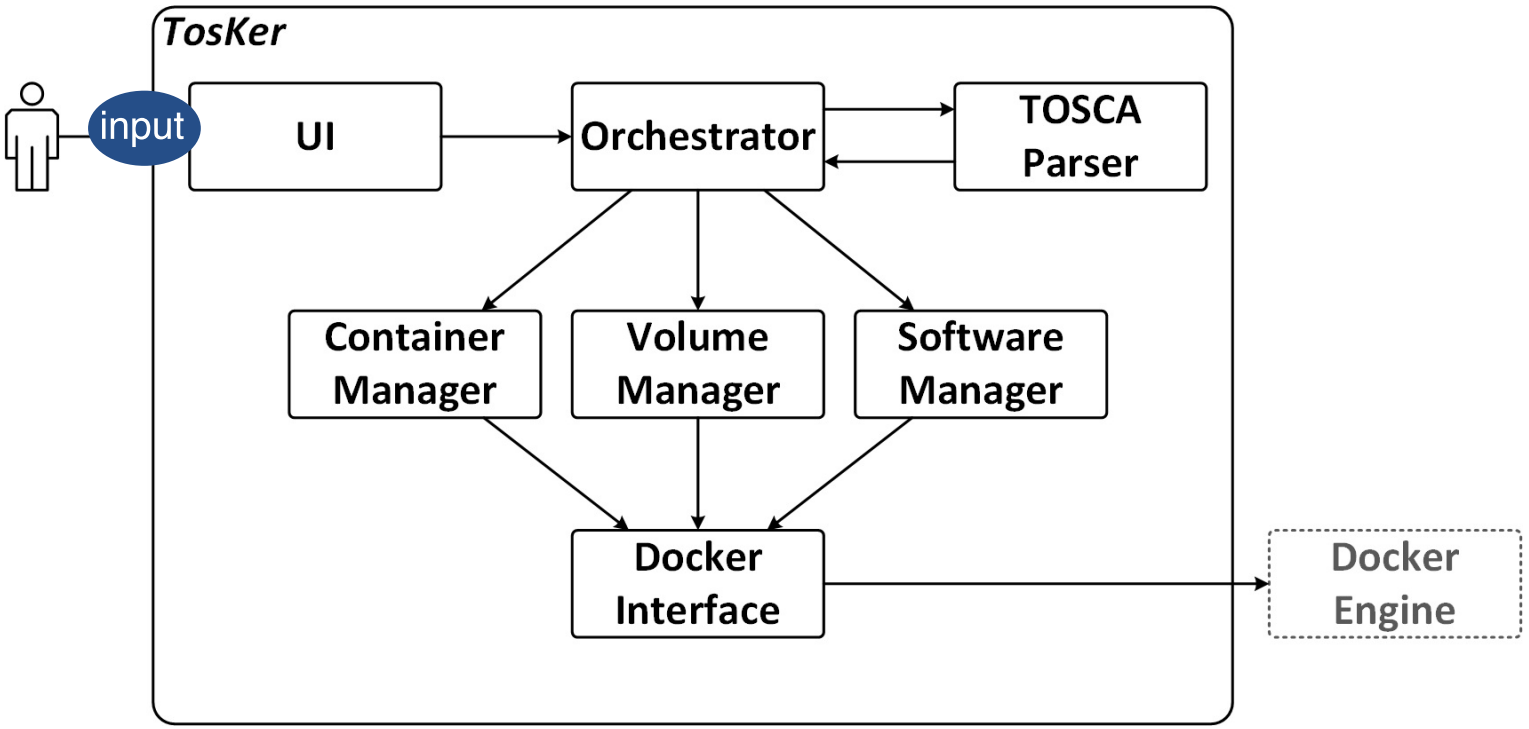
Case study: Thoughts



Architecture



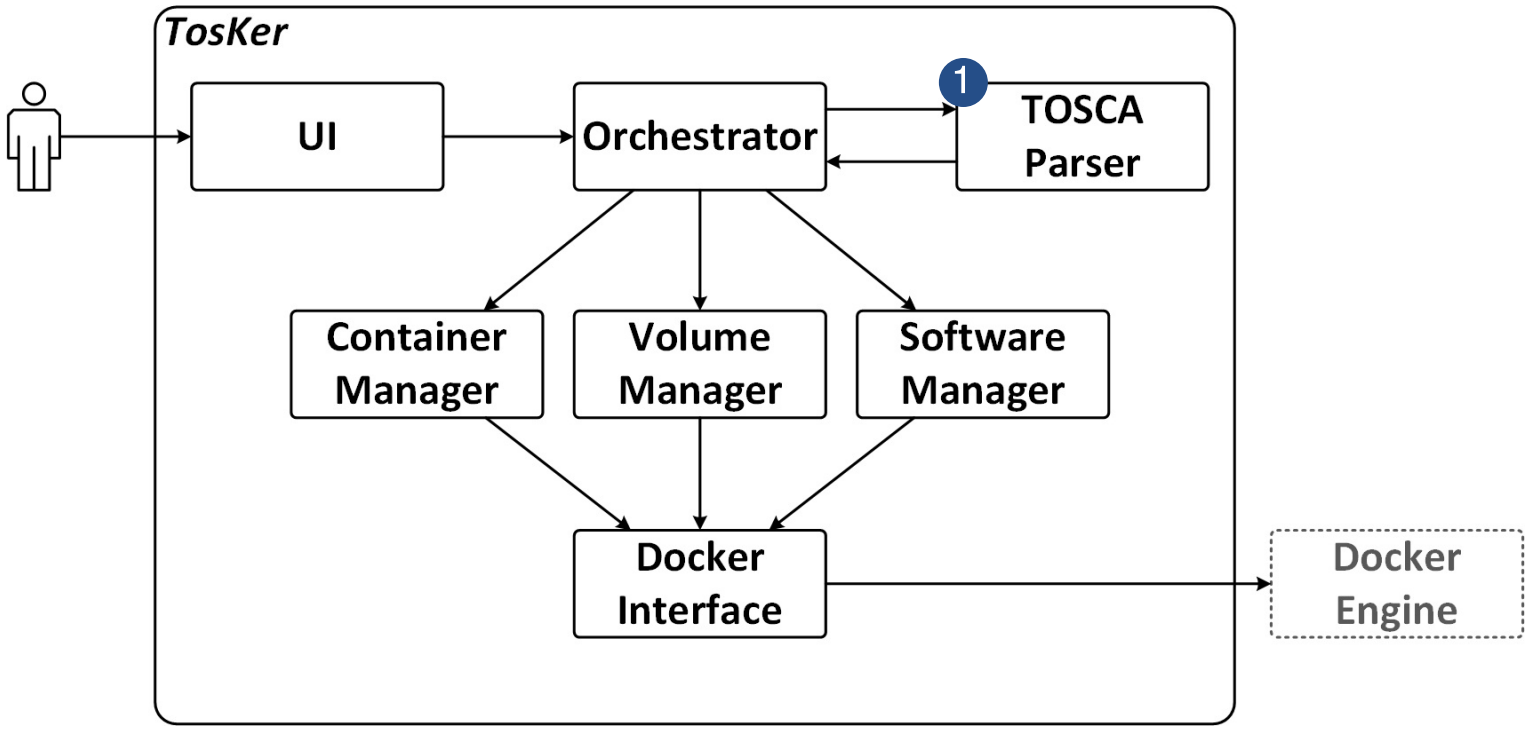
How TosKer orchestrates applications



The input of TosKer is

- a TOSCA application specified using TosKer types, and
- management operation(s) to perform.

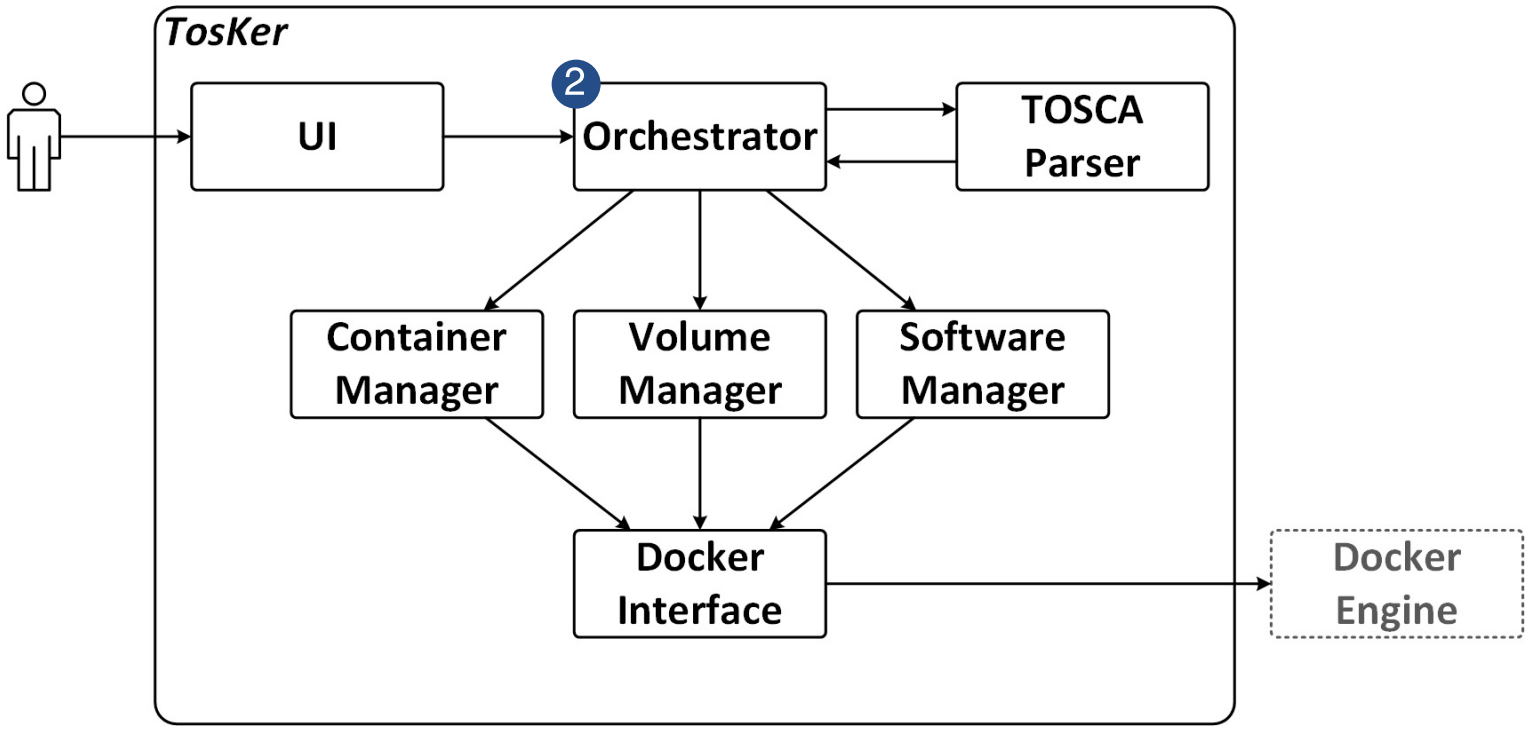
How TosKer orchestrates applications



TosKer

- parses and validates the TOSCA application, and
- executes a topological sorting algorithm.

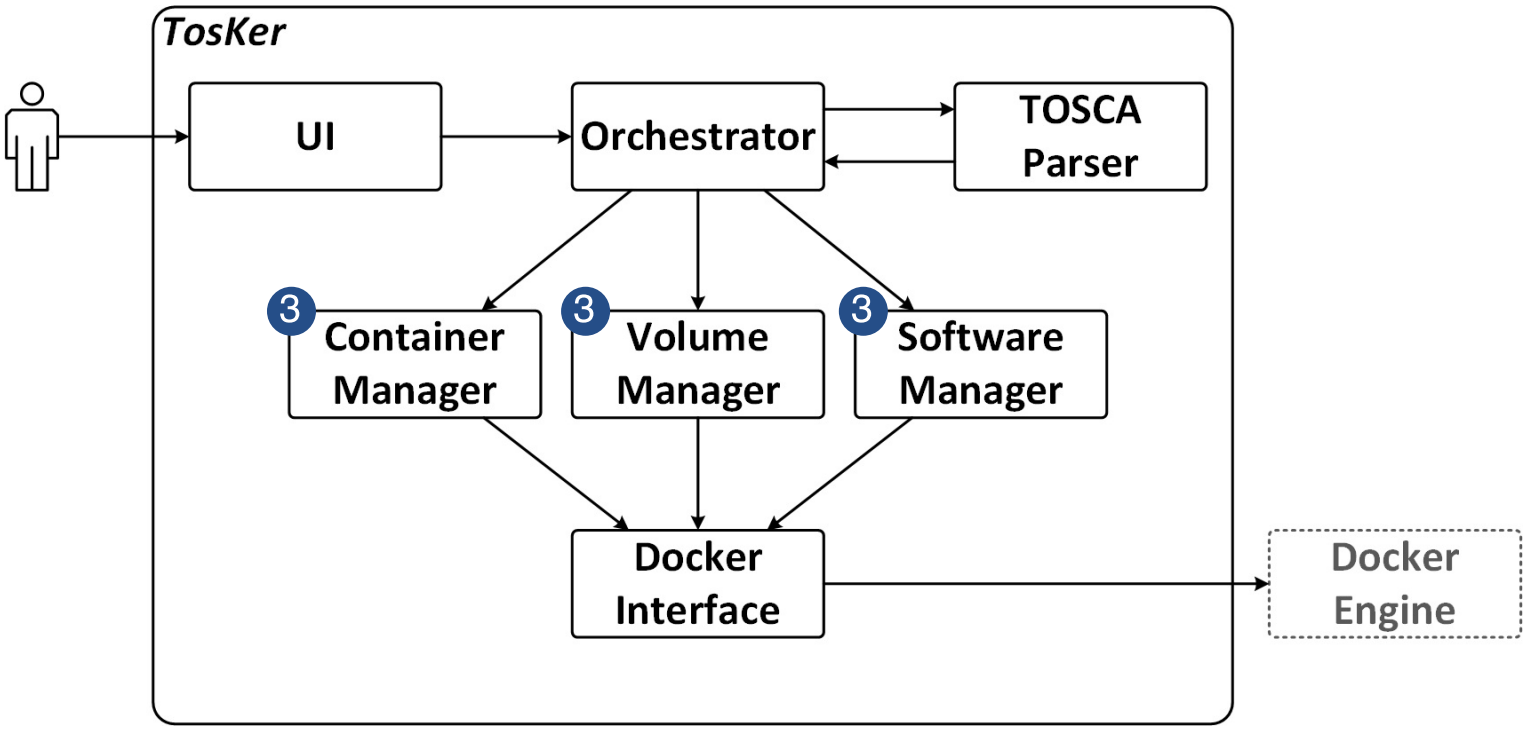
How TosKer orchestrates applications



TosKer

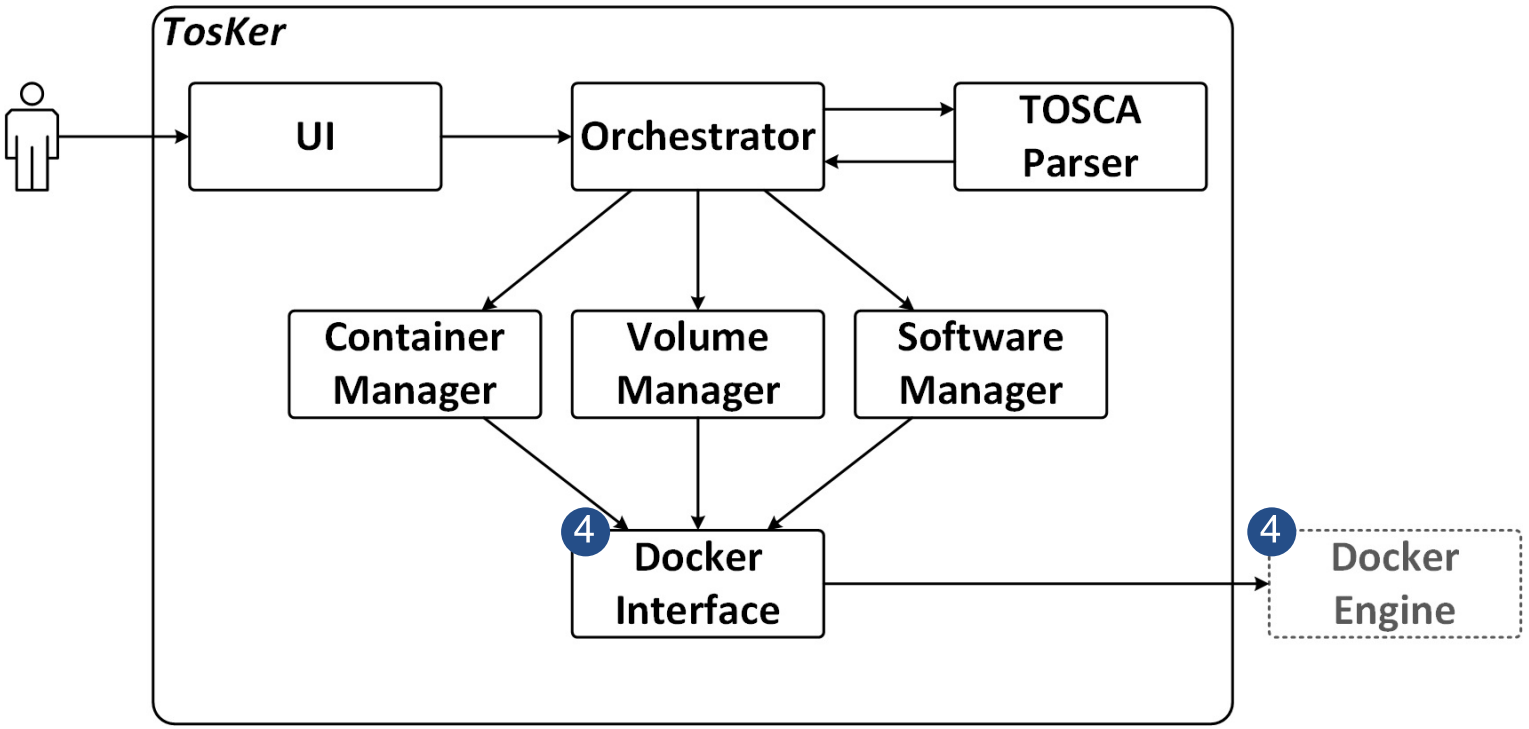
- scans the sorted application topology, and
- for each component, it calls a specific operation (e.g., create)

How TosKer orchestrates applications



Each manager is in charge of implementing/executing the invoked operation on a component...

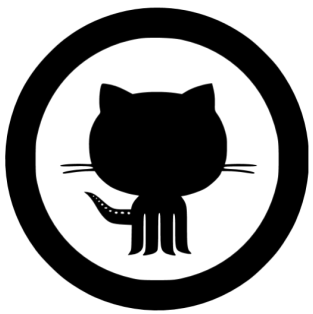
How TosKer orchestrates applications



...by properly invoking the Docker engine (through the Docker interface)

Implementation

Python



GitHub

MIT Licence

- **PyPI**: https://pypi.python.org/pypi/tosKer
  `pip install tosker`
- **GitHub**: https://github.com/di-unipi-socc/TosKer

Advantages and Limitations

Table of Contents

Future works

Conclusions

Thank You

Q&A