Combining TOSCA and Docker

Luca Rinaldi

University of Pisa

June 2017

Table of Contents

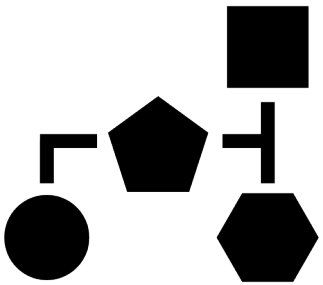
Table of Contents

Software deployment and Orchestration

?

Software deployment and Orchestration

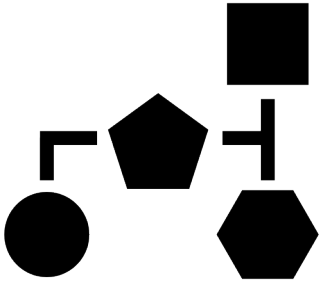
?



Application specification

Software deployment and Orchestration

?

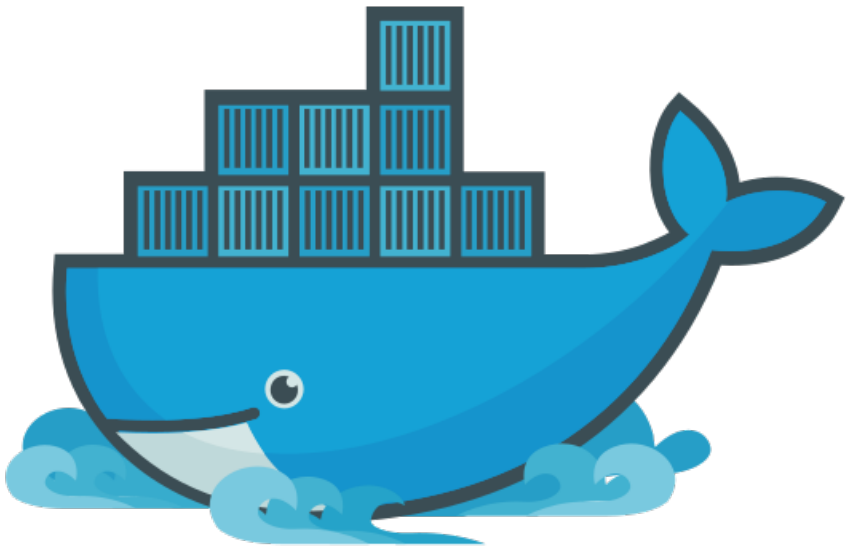


Application specification

Application orchestration

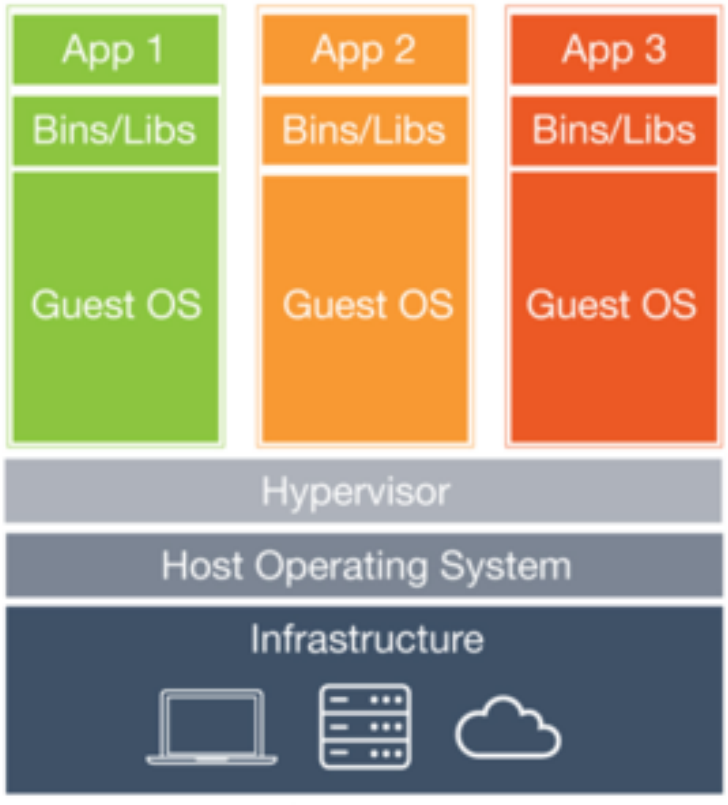
Table of Contents

Docker

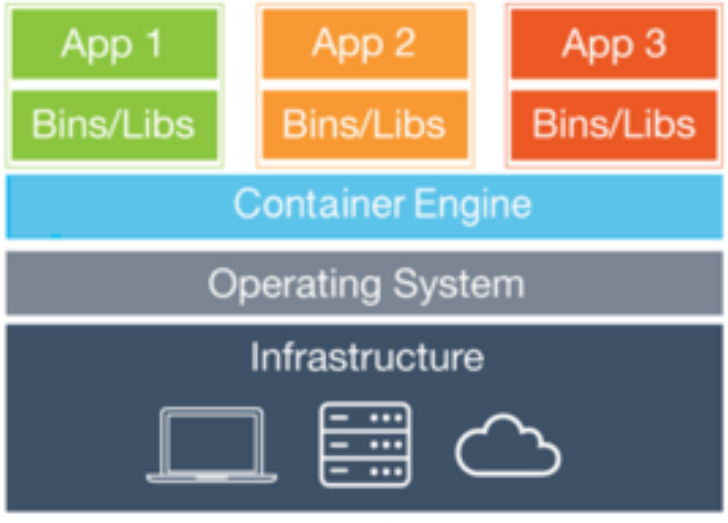


Docker is a tool that can package an application and its dependencies in a virtual container that can run on any Linux server.

VM vs Docker

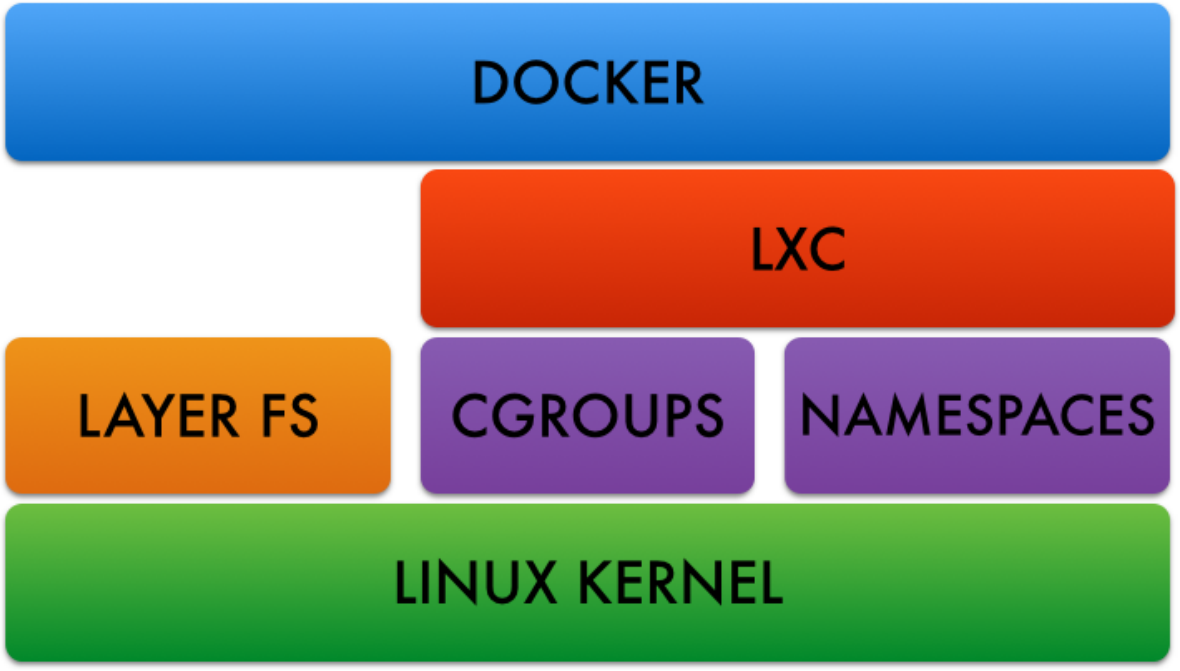


Hypervisor-based Virtualization



Container virtualization

Docker Architecture



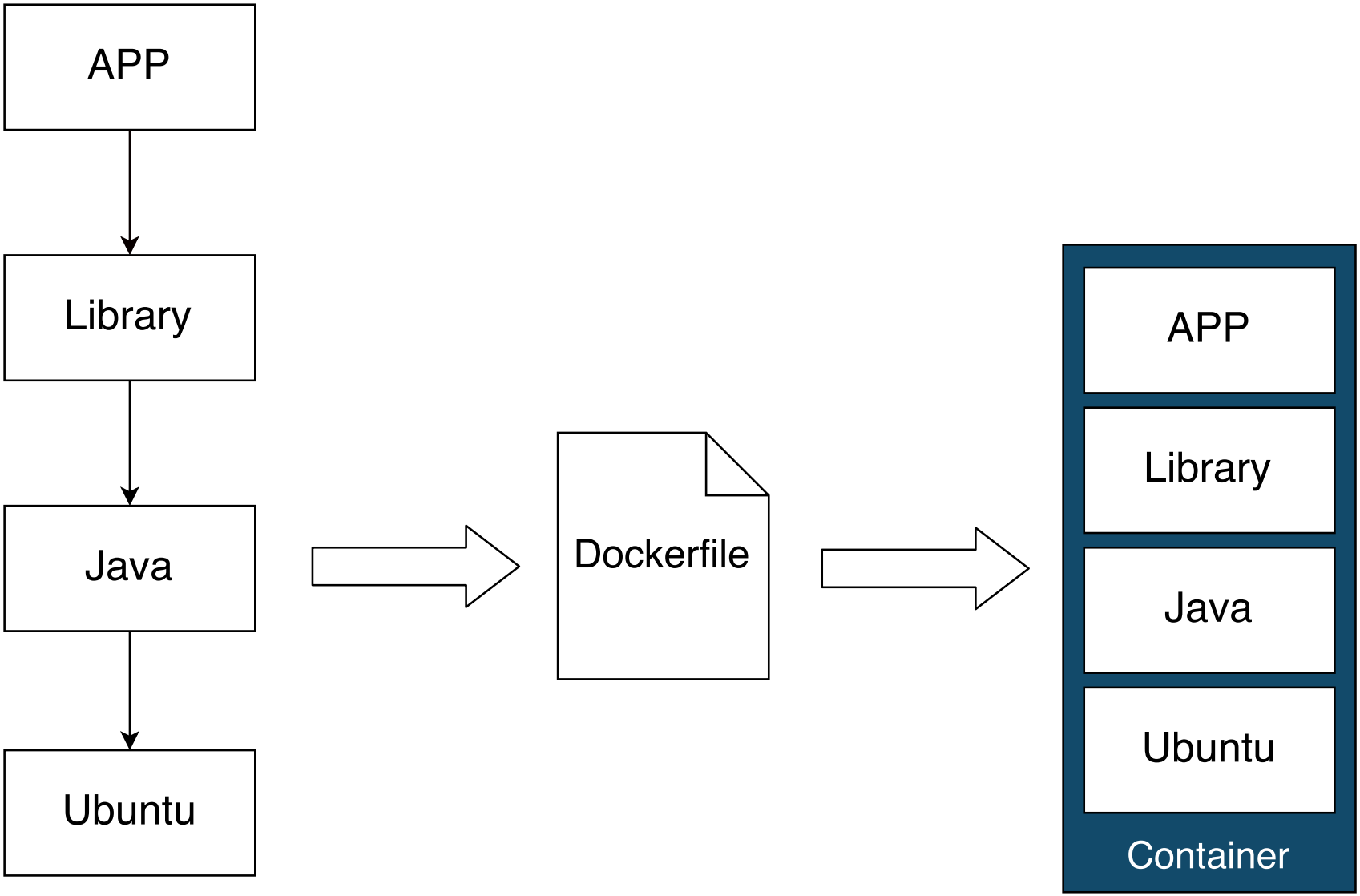
- **LXC**, an operating-system-level virtualization method
- **Layer file-system**, a Union file system (which implements: `combine updates` and `copy-on-write`)

Docker main concepts

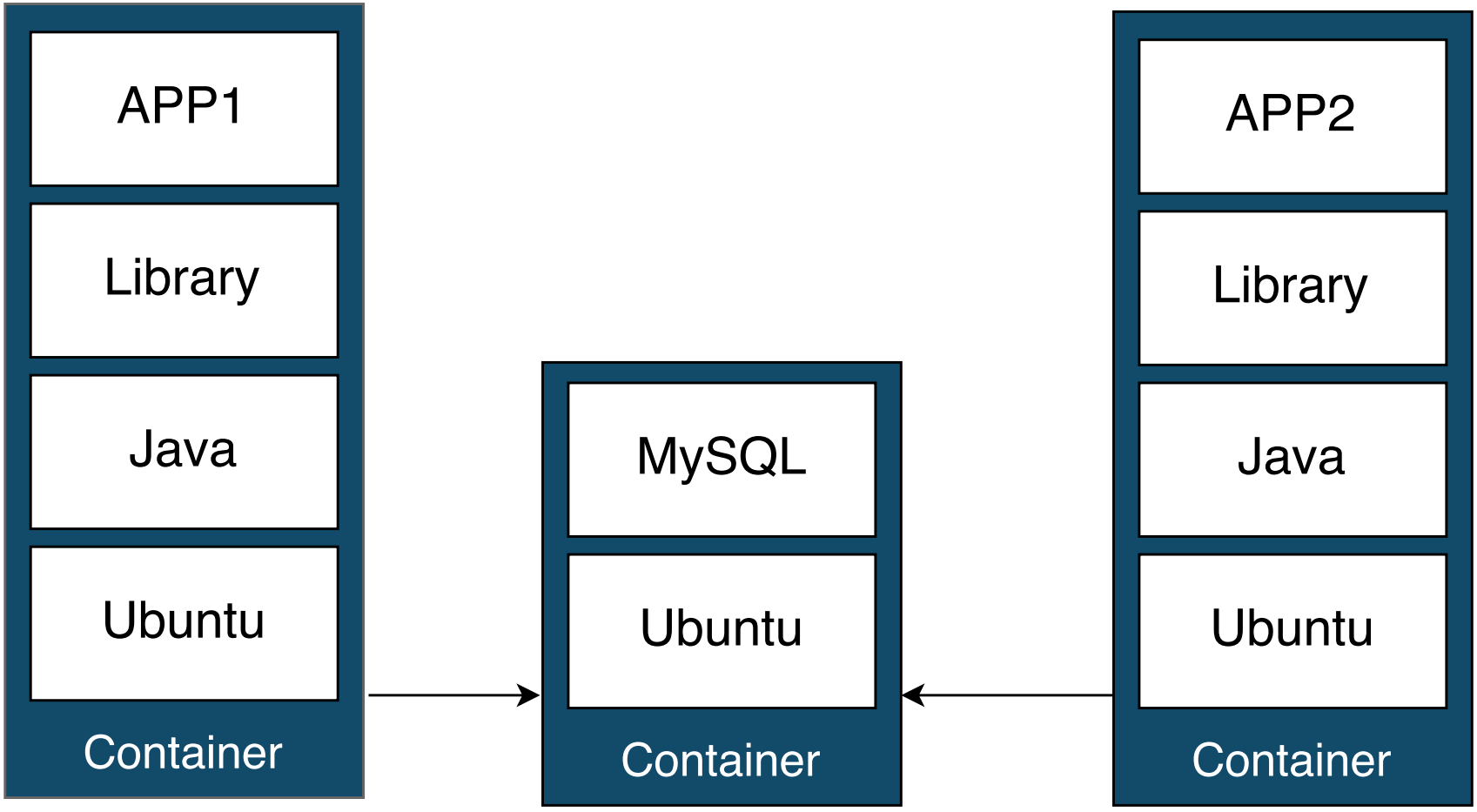
Main concepts of the Docker platform

- **Dockerfile**, a script to generate an Image
- **Docker Image**, a separated file-system with all the binaries and library
- **Docker Container**, running instance of a Docker Image
- **Docker Volume**, a persistent data storage
- **Docker Hub**, a public database of Docker Images

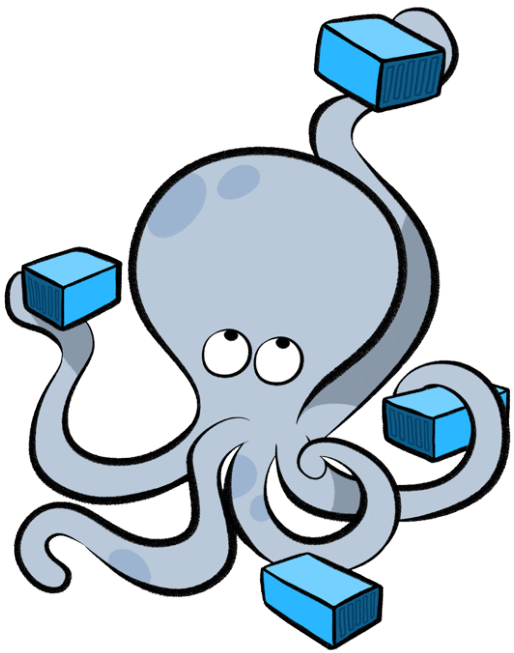
How use Docker



Multi-container application



Docker Compose



```
version: "2"
services:
  web:
    build: .
    ports:
     - "5000:5000"
    volumes:
     - .:/code
  redis:
    image: "redis:alpine"
```

Problems

- The container hide the components that it contains

Problems

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

Problems

- The container hide the components that it contains
- Poor application orchestration
- Can orchestrate only container (Docker compose)

Table of Contents

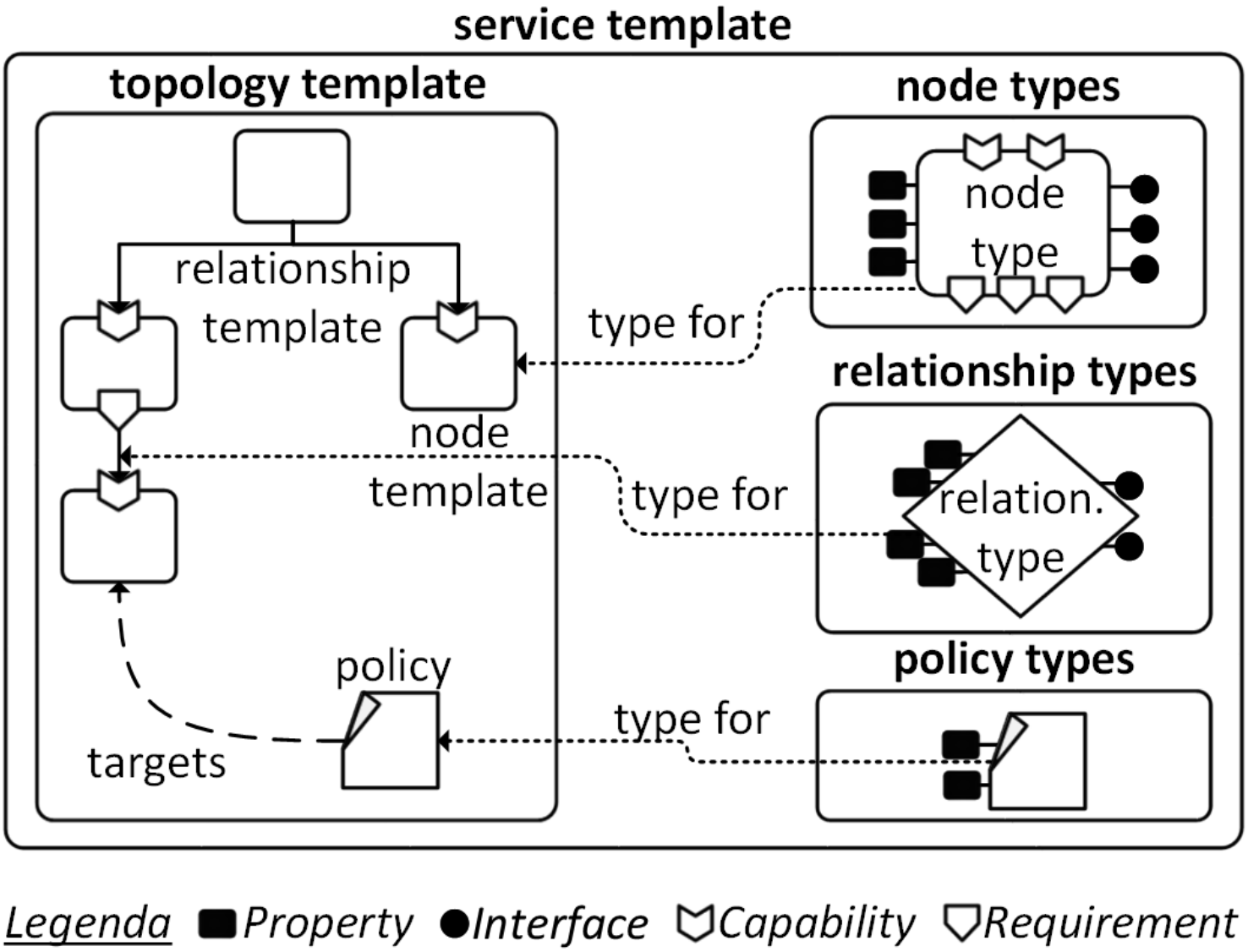
TOSCA

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

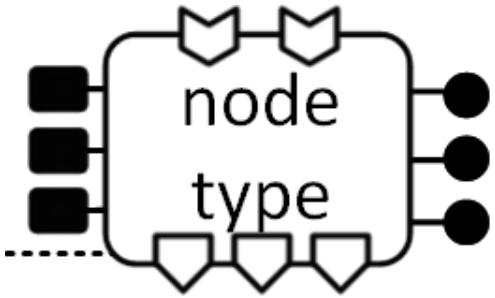
Main concepts

- The description use YAML
- CSAR it is an archive with the TOSCA description and all the artifacts
- declarative processing

TOSCA description

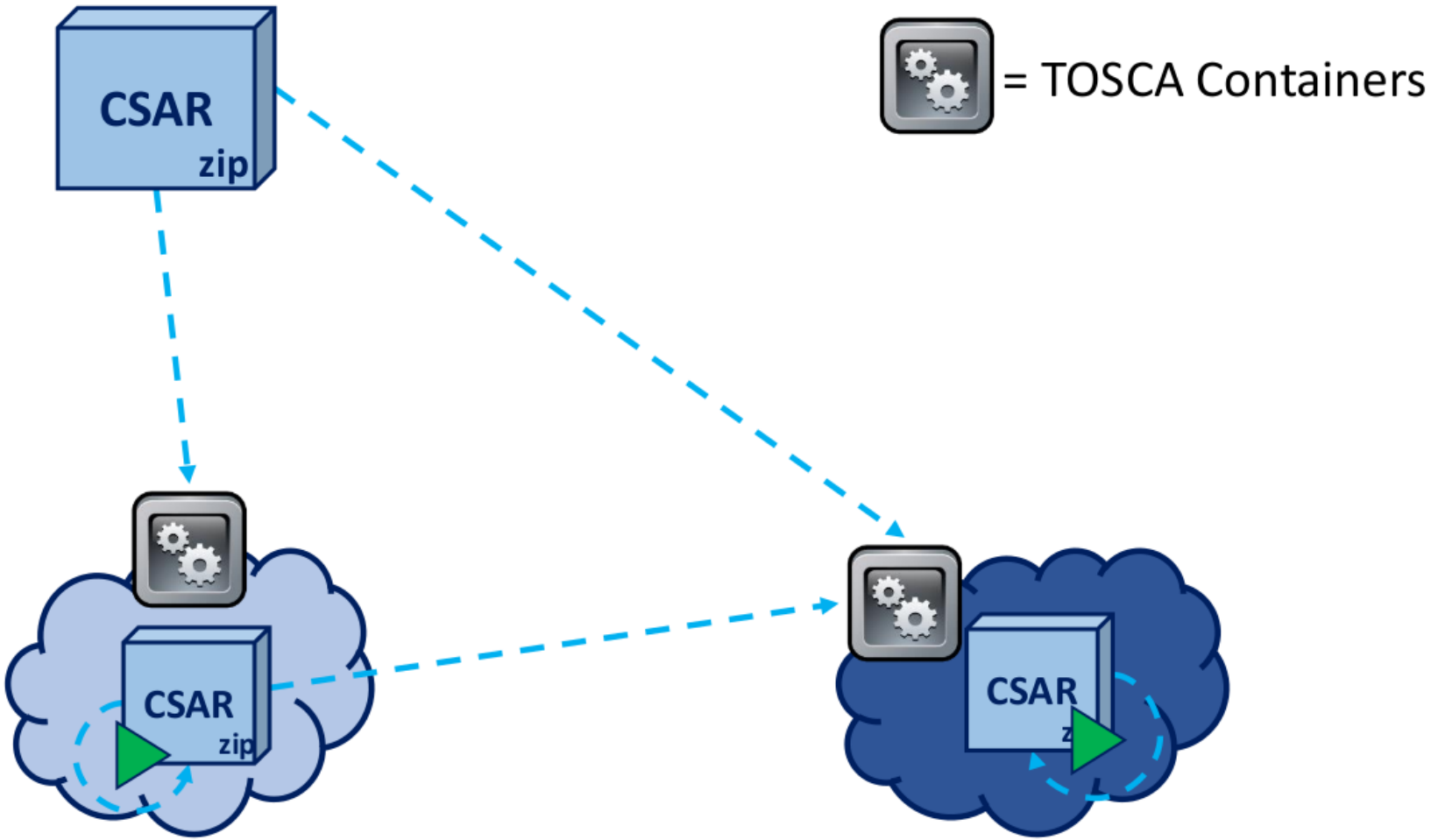


Node type

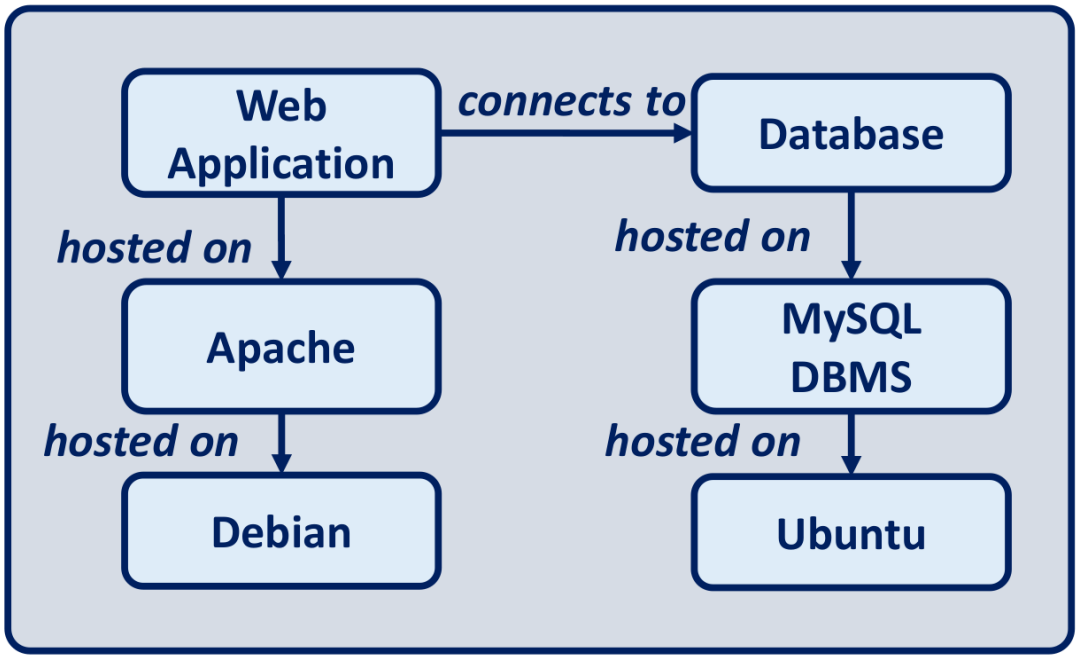


- **requirements**, what node requires
- **capabilities**, what node offers
- **properties**, properties of the node
- **interfaces**, operations to deploy and manage the node
- **artifacts**, data needed by the node to implement the interface

How it works

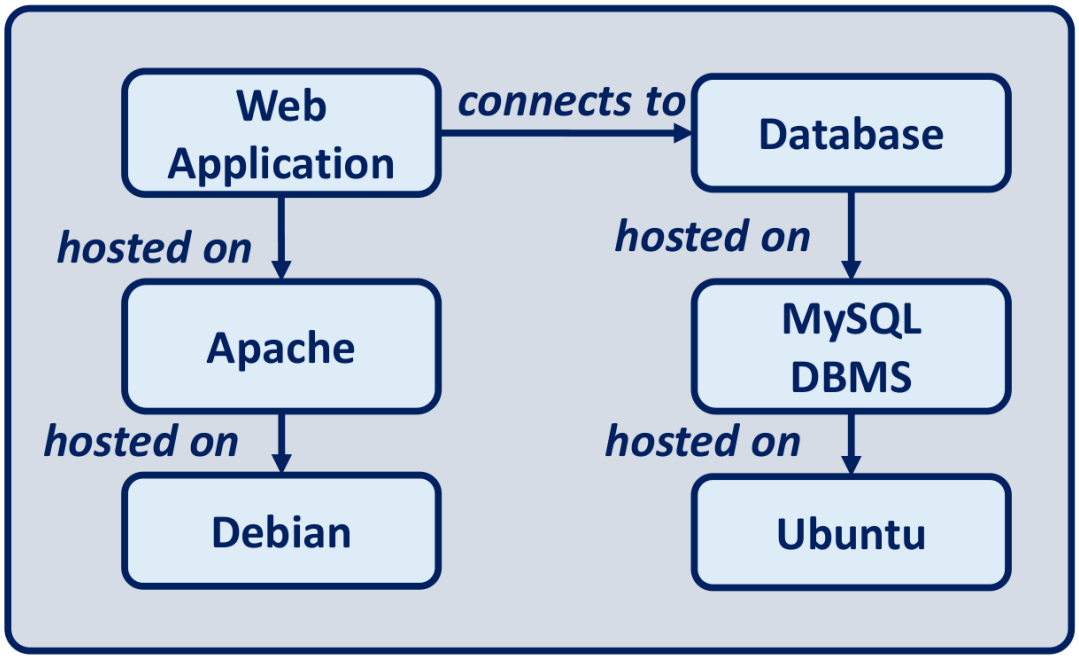


Declarative process



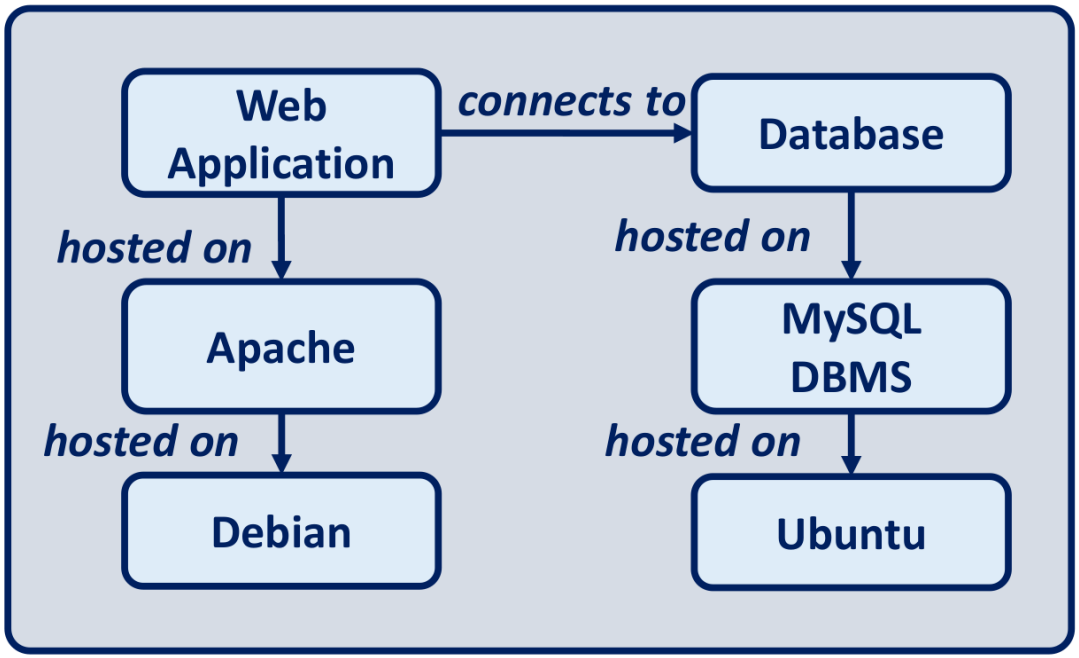
1. Debian
2. Ubuntu

Declarative process



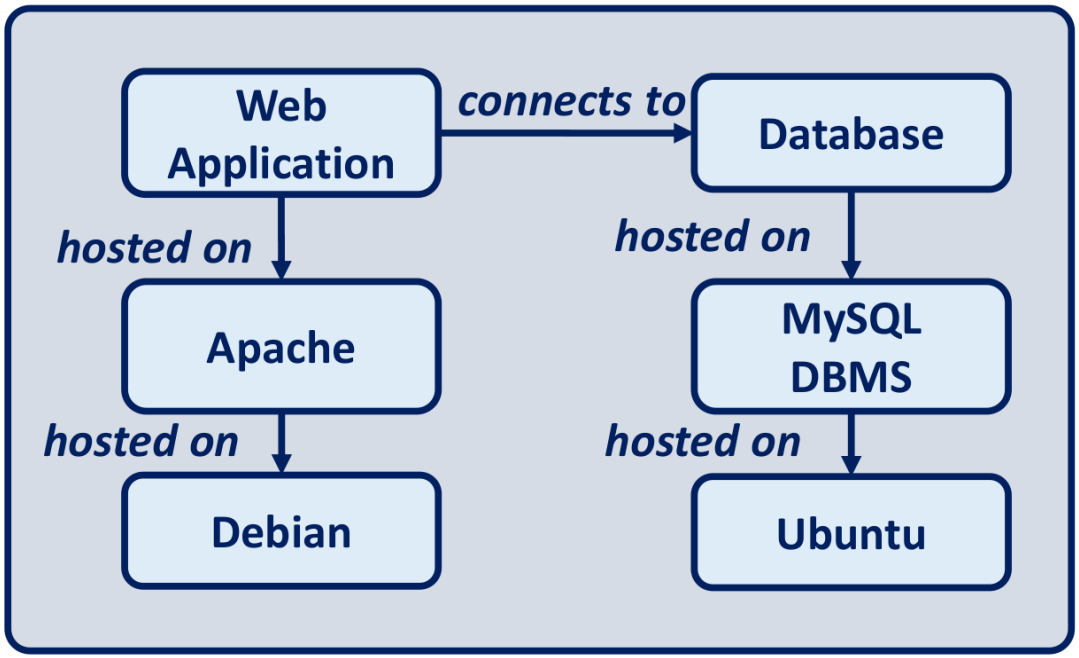
1. Debian
2. Ubuntu
3. Apache
4. MySQL DBMS

Declarative process



1. Debian
2. Ubuntu
3. Apache
4. MySQL DBMS
5. Database

Declarative process



1. Debian
2. Ubuntu
3. Apache
4. MySQL DBMS
5. Database
6. Web Application

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.

- inputs a TOSCA description of a multi-component application
- automatically deploys and orchestrates it using the Docker engine

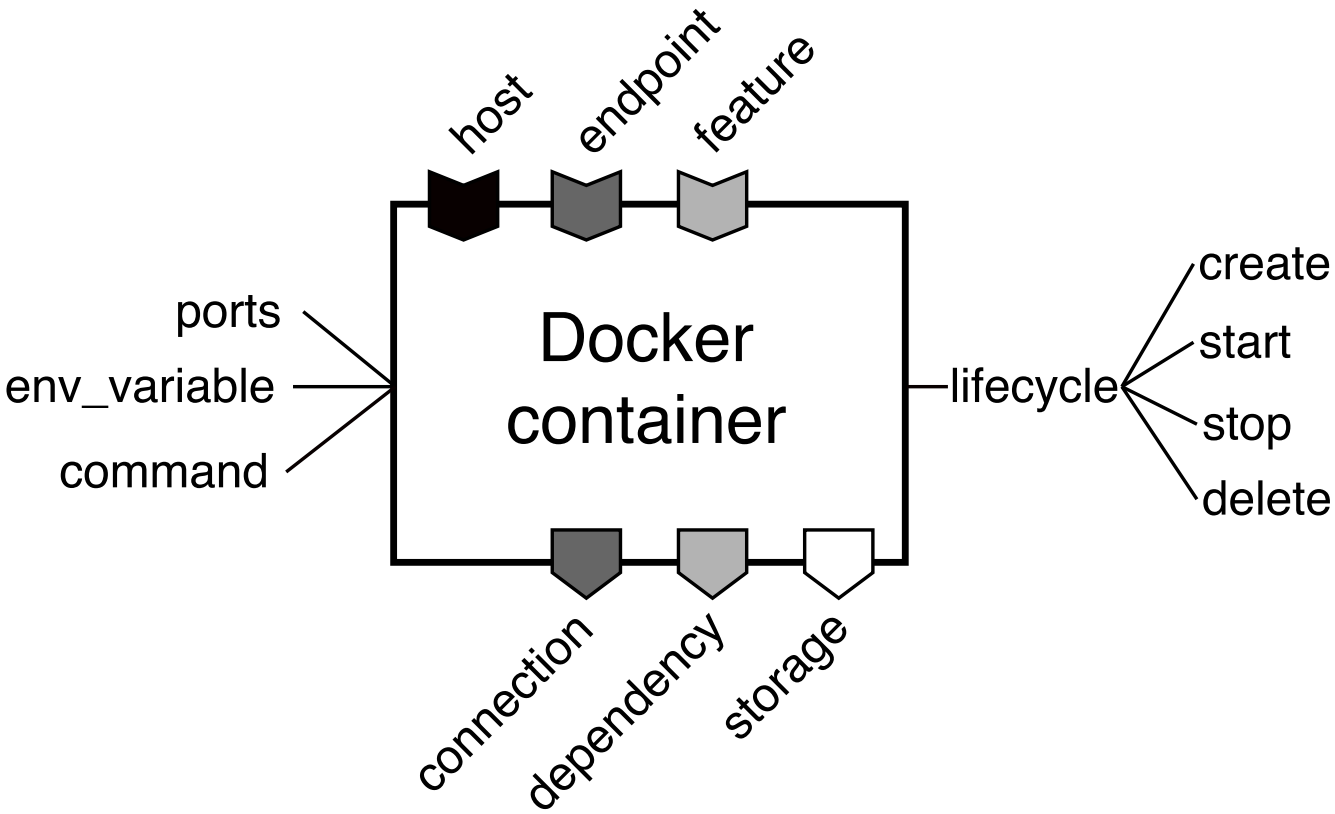
Describing applications with TosKer

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

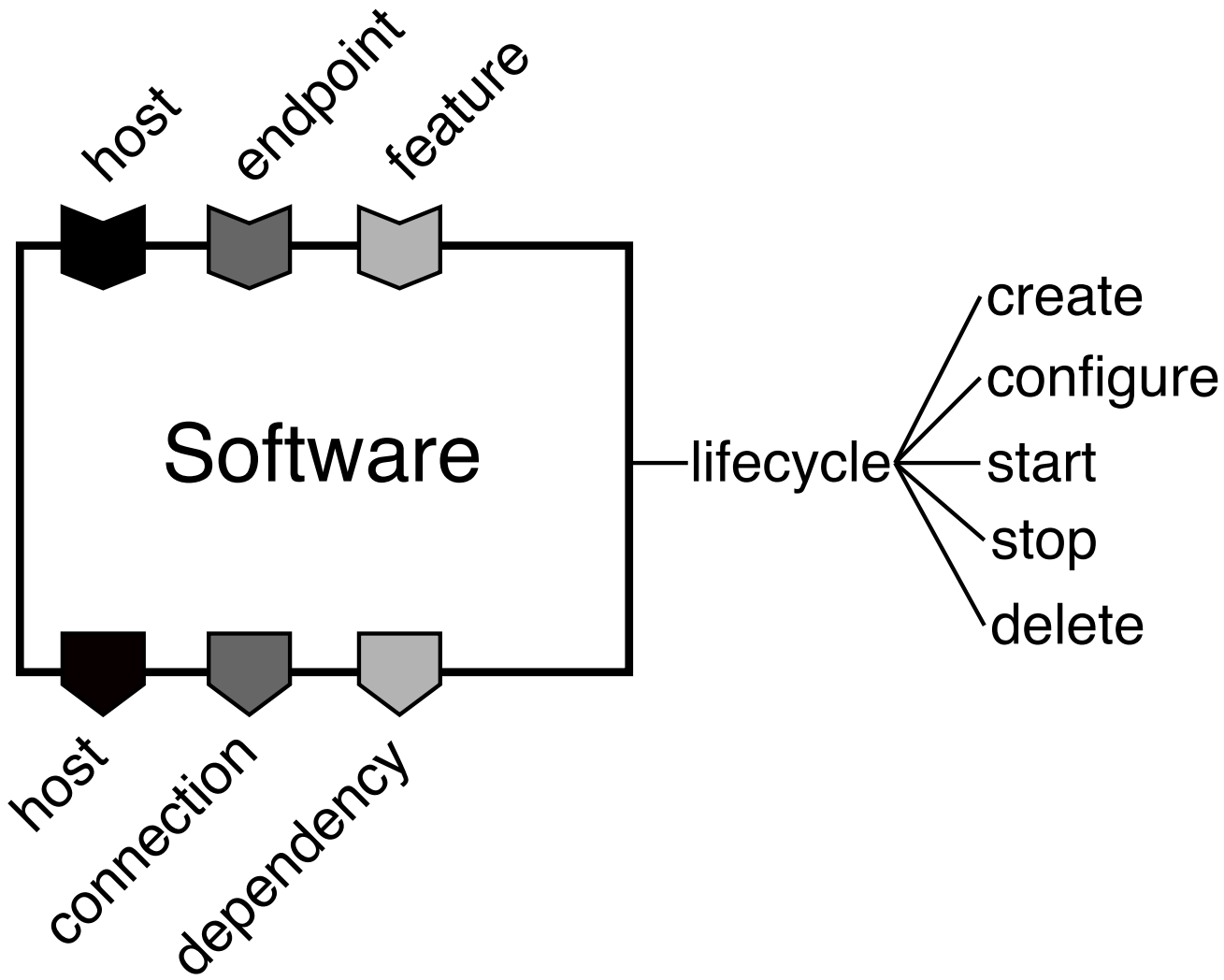
Describing applications with TosKer

- 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`)

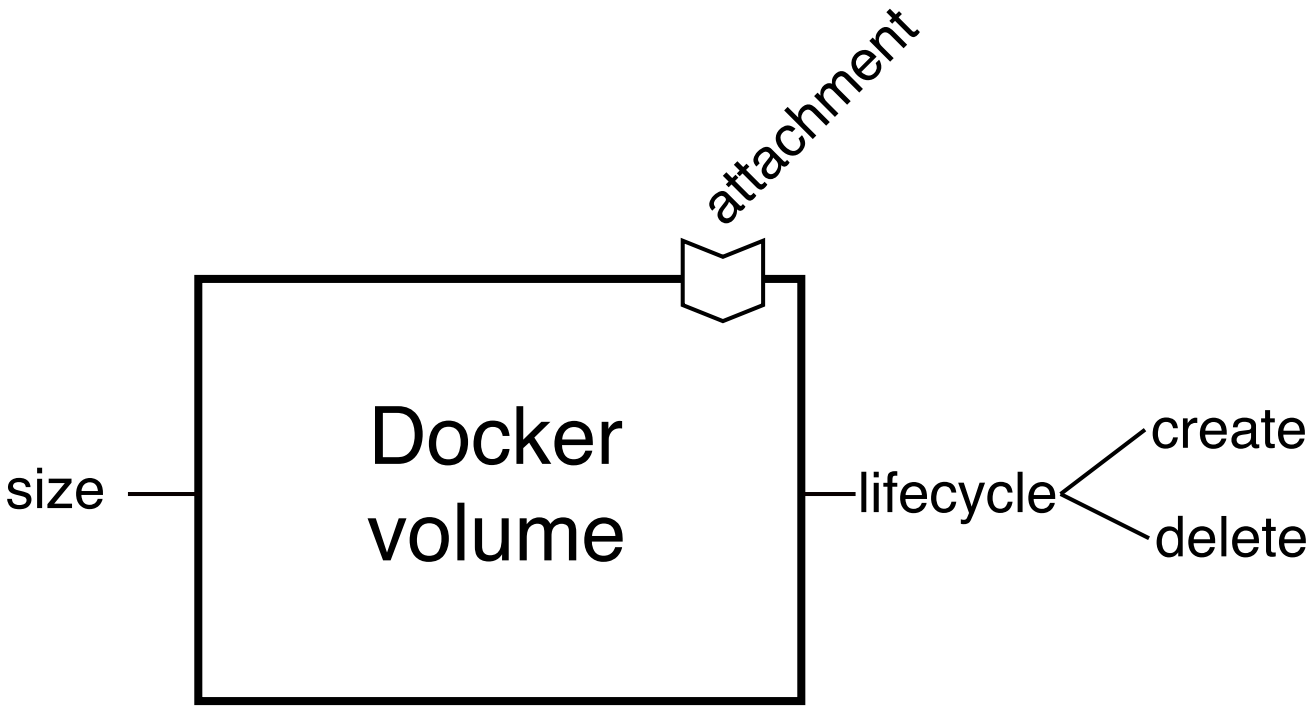
tosker.nodes.Container



tosker.nodes.Software



tosker.nodes.Volume

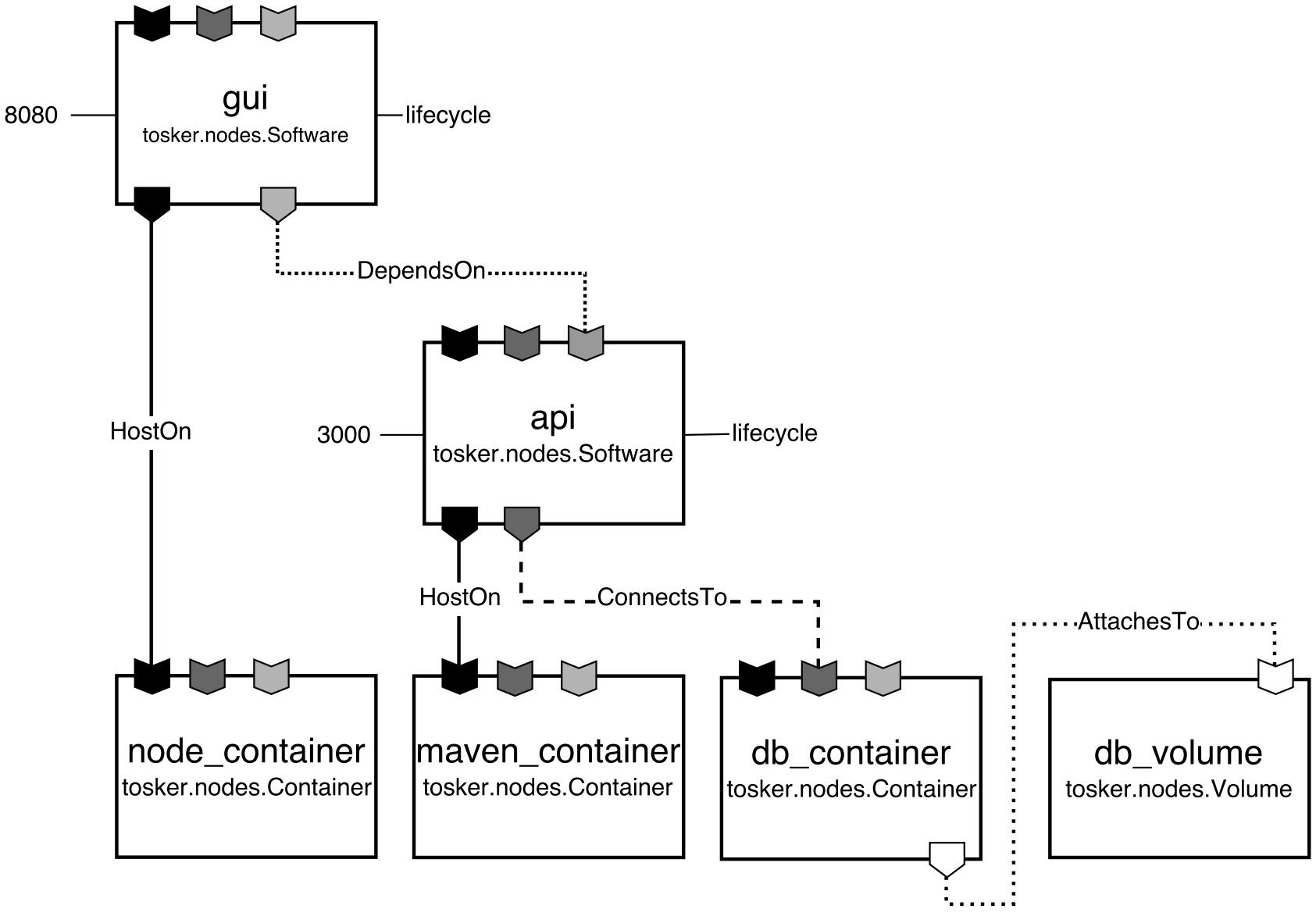


Constrains on the composition

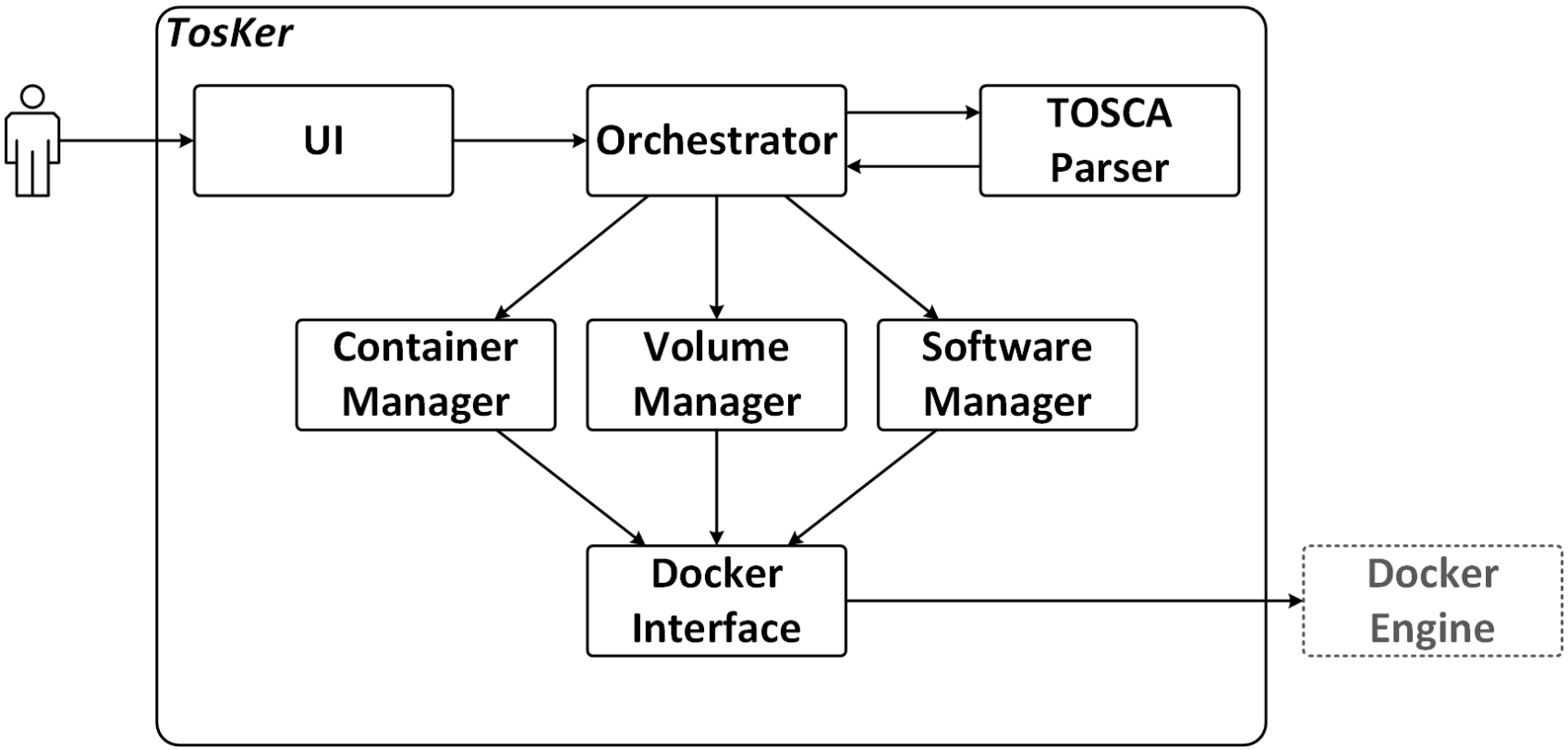
Each application must meet some constrains, e.g.,

- 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*

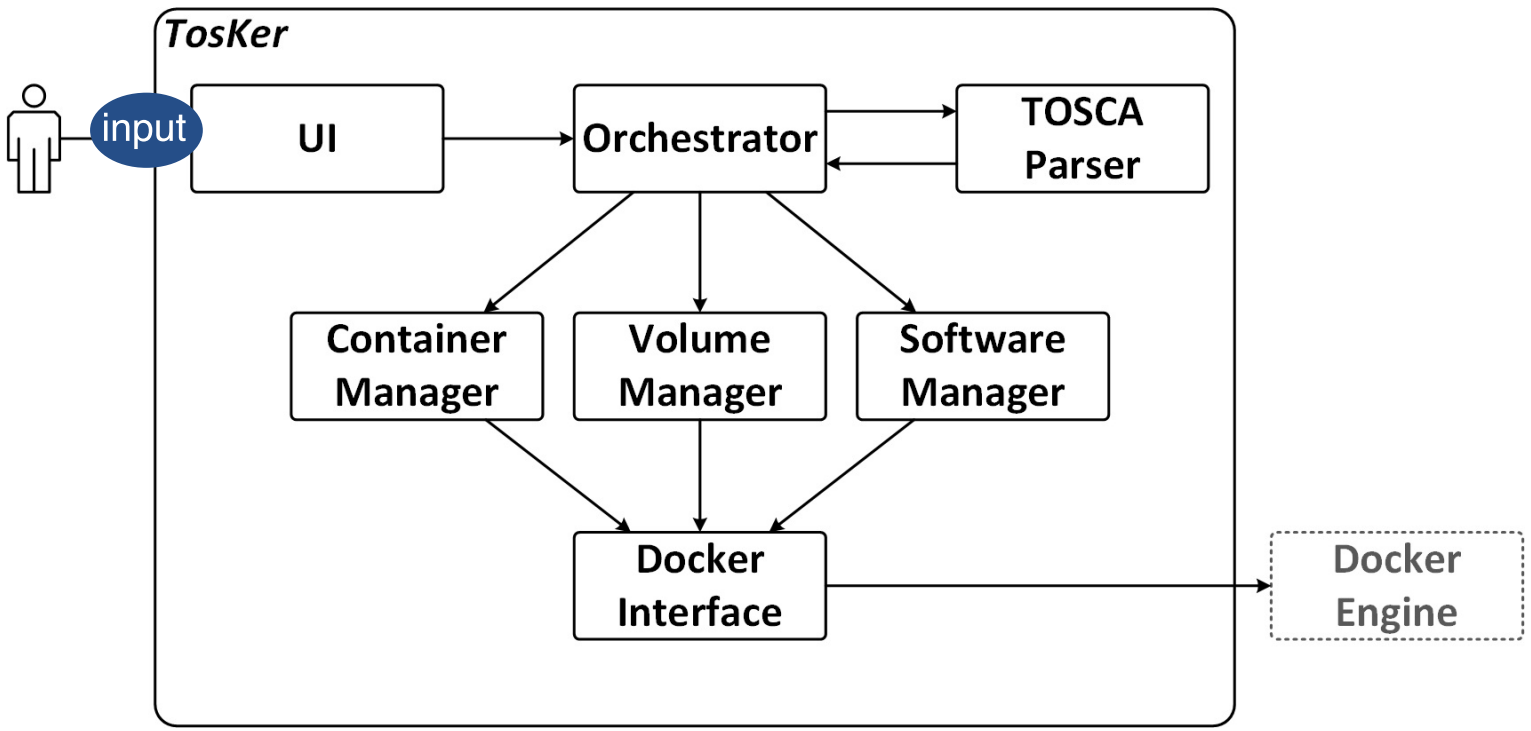
Case study: Thoughts



Architecture



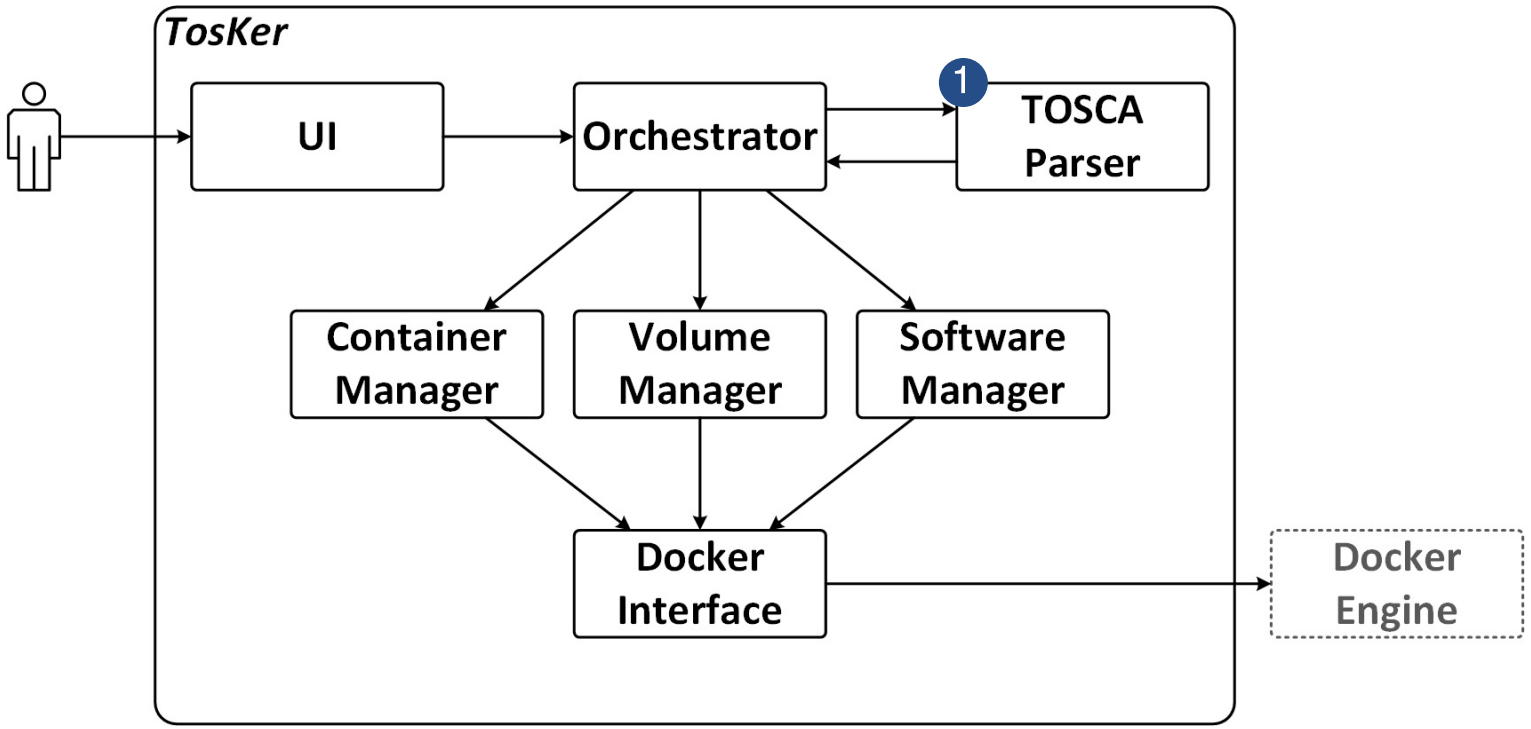
How TosKer orchestrates applications



The input of TosKer is

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

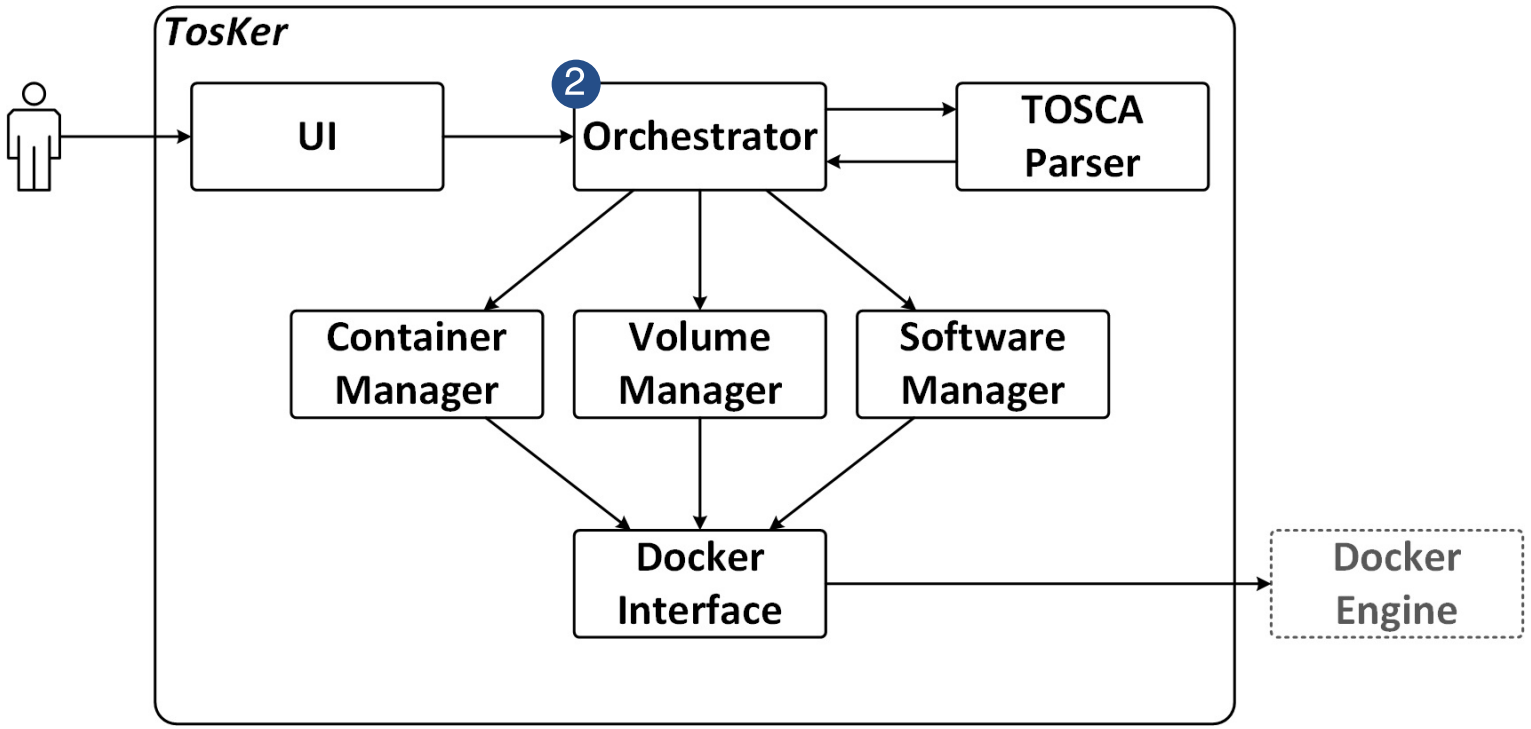
How TosKer orchestrates applications



TosKer

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

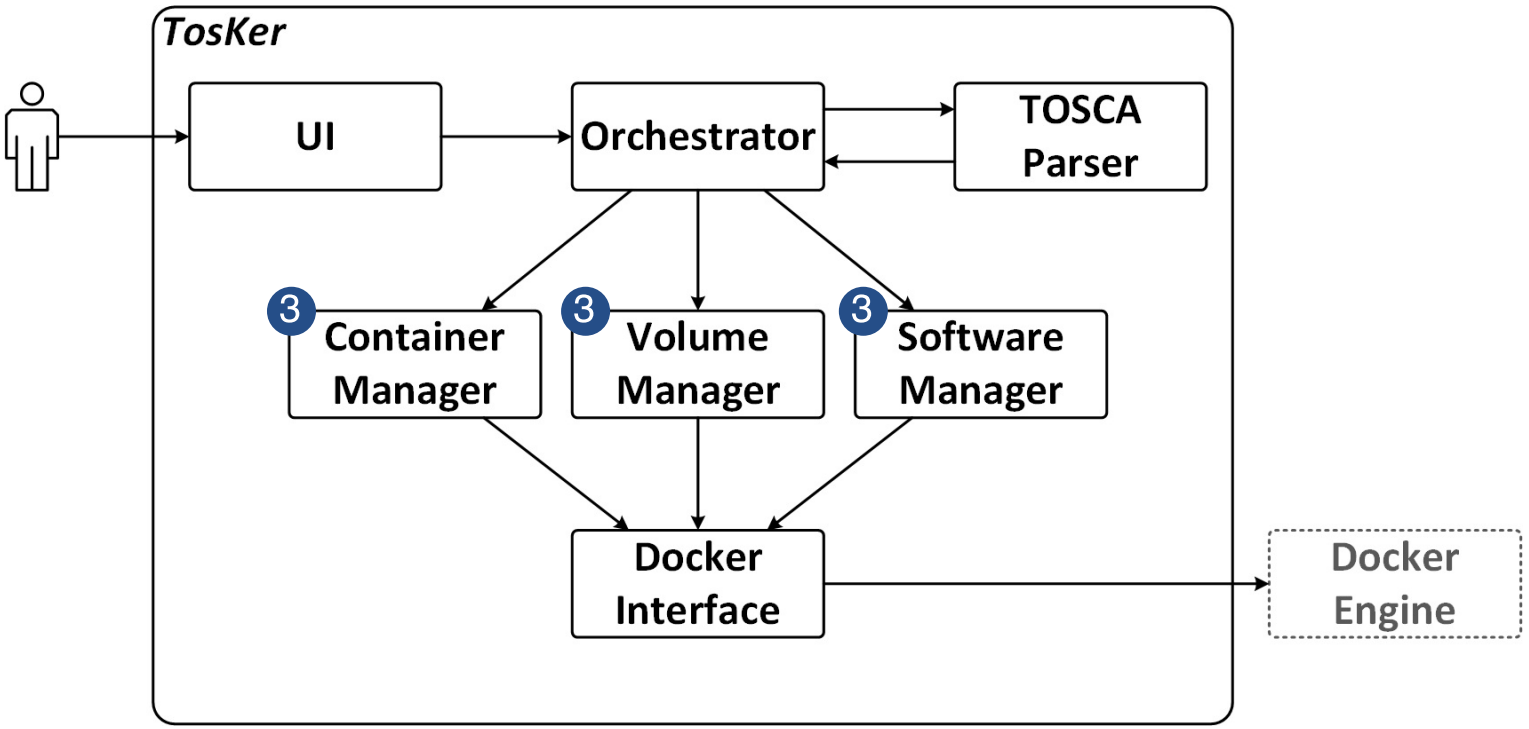
How TosKer orchestrates applications



TosKer

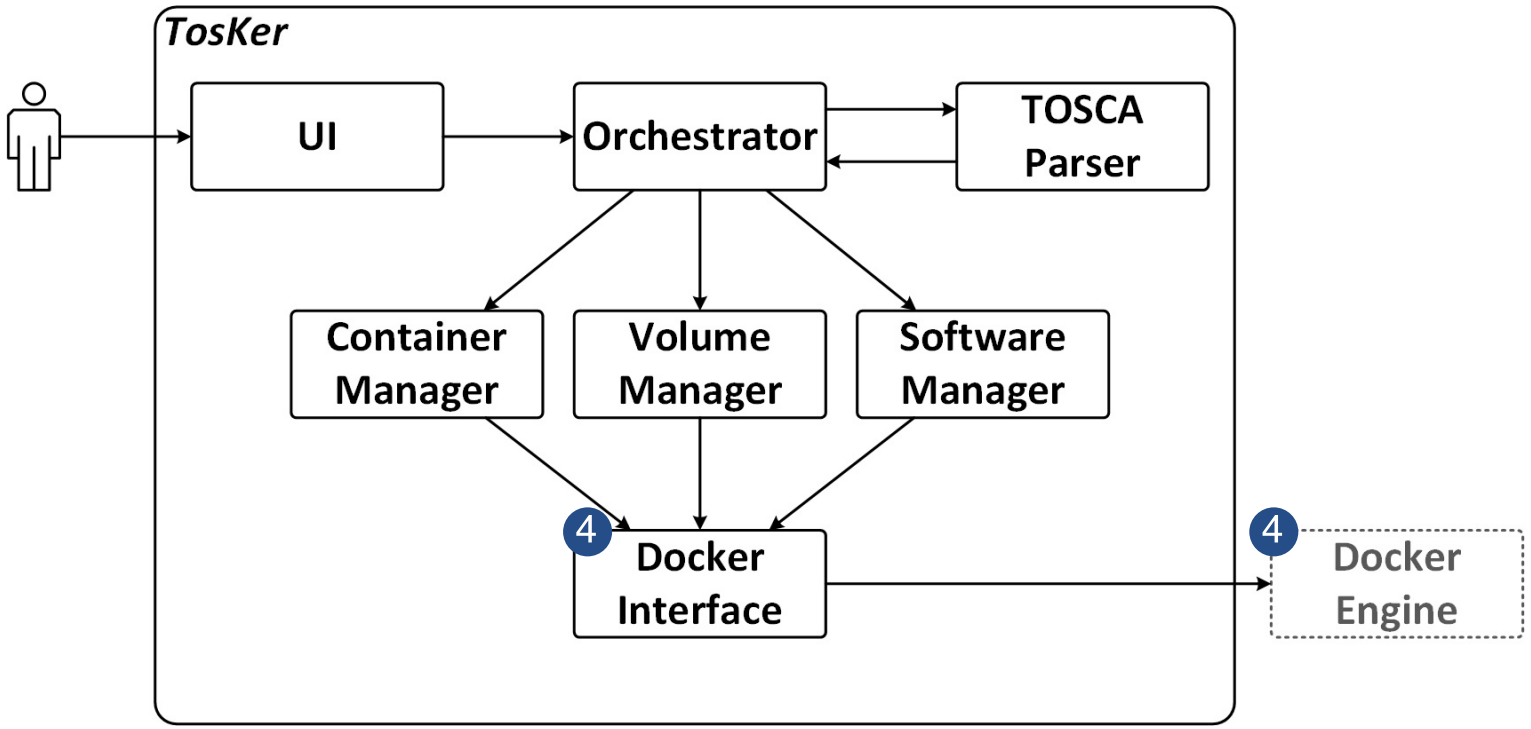
- scans the sorted application topology, and
- for each component, it calls a specific operation (e.g., create)

How TosKer orchestrates applications



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

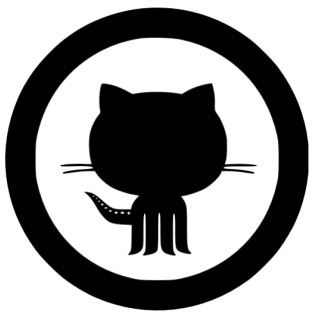
How TosKer orchestrates applications



...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

Table of Contents

Conclusions

On the context of Deployment and Orchestration application, we treat:

- Docker
- TOSCA YAML

Conclusions

On the context of Deployment and Orchestration application, we treat:

- Docker
- TOSCA YAML

Then we present the TosKer orchestration engine, which

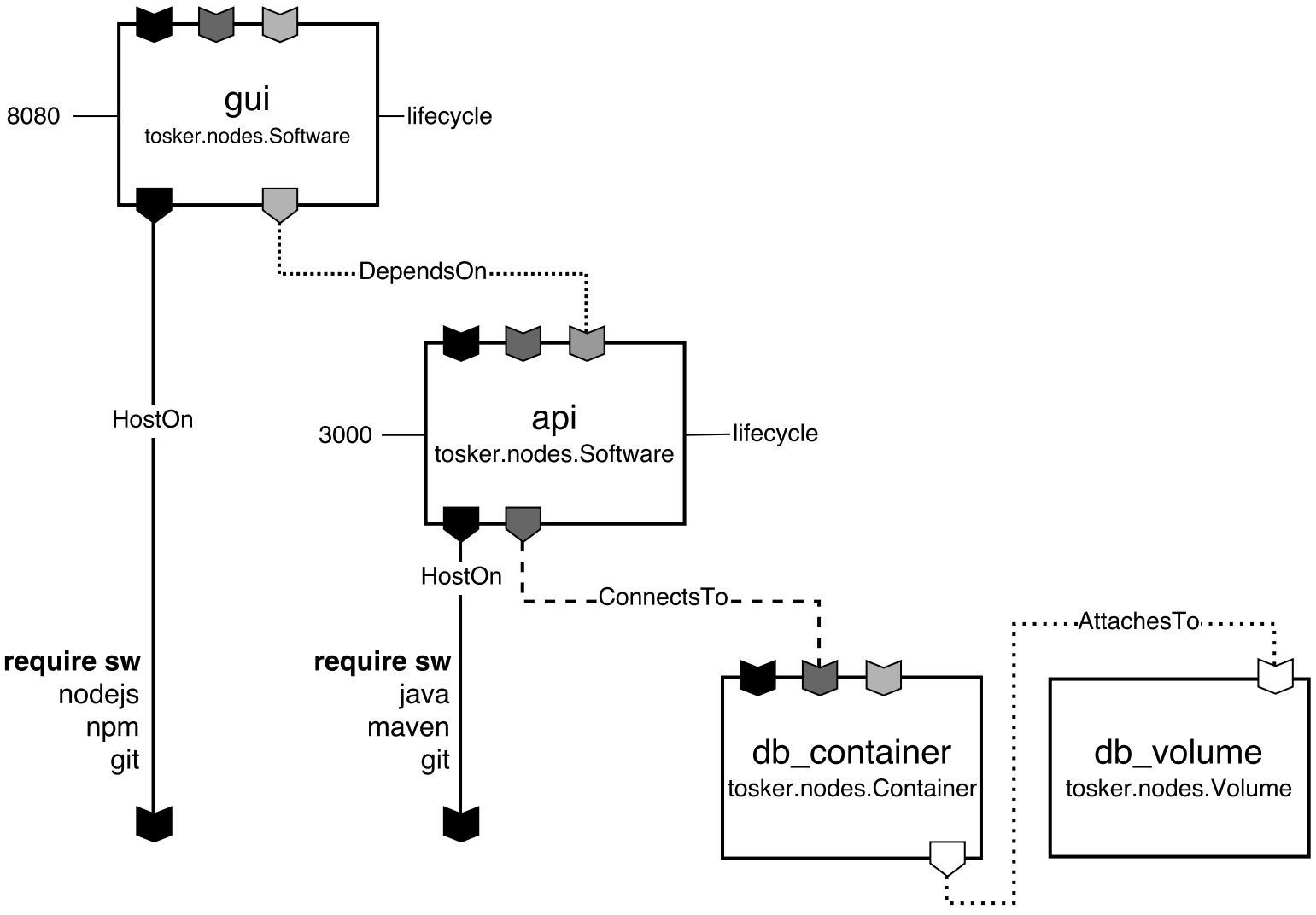
- extends TOSCA by providing a Docker-based orchestration engine for TOSCA application, and
- extends Docker by adding the capability of orchestrating software components together with Docker containers/volumes

Imminent work

Dockerize, a completer of TosKer specification.

Imminent work

Dockerize, a completer of TosKer specification.



Future work

- Automatically determine the Docker containers needed to effectively run an application (**Dockerize**)
- Support cluster of workstations and external cloud services
- Integrate TosKer with fault-aware management protocols

Thank You

Q&A