Container-based deployment of ROS applications

Mathias Lüdtke, Fraunhofer IPA

Structure

- Deployment?
- Container?
- Overview IPDE framework
- Current state
- Outlook

Deployment

- Many definitions with varying scopes..
- Turning source code into running systems
  - Set-up environment on target platform
  - Transfer source code and config
  - Install dependencies
  - Build source code
  - Run application
- Focus on ROS-based applications

ROS developer's workflow

1)Install ROS

2)Prepare catkin workspace

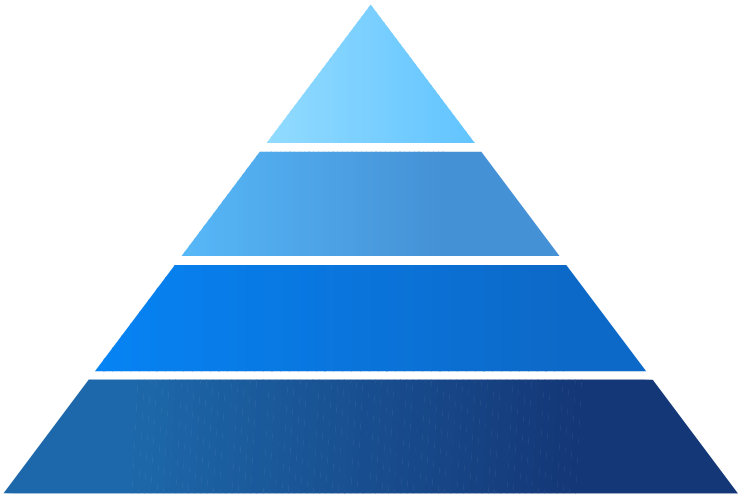
3)git clone, rosinstall

4)rosdep install

5)catkin_make

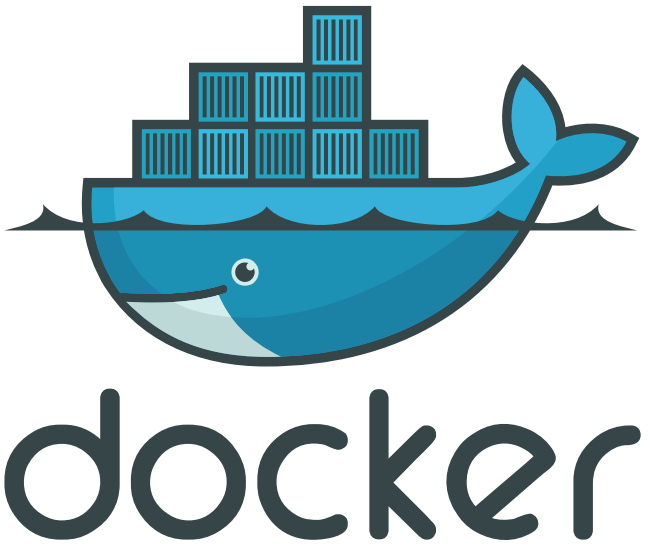
6)roslaunch ..

7)profit



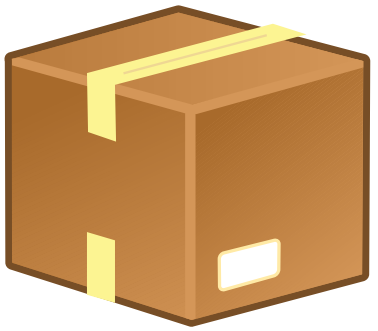
Software Containers

- Operating-system-level virtualization
- “Light-weight VMs“
- Shipped with Linux mainline kernel
- On top: Docker
  - Container engine
  - Overlay filesystems
  - Docker Hub
  - Cross-platform Tooling



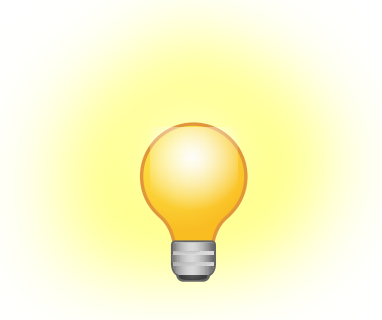
How does it help?

- Self-contained images
- Images can be run ad-hoc, no booting
- Minimal overhead
- Standardized environment
- Host isolation
- Cloud-ready



Integration Platform & Deployment Environment (IPDE)

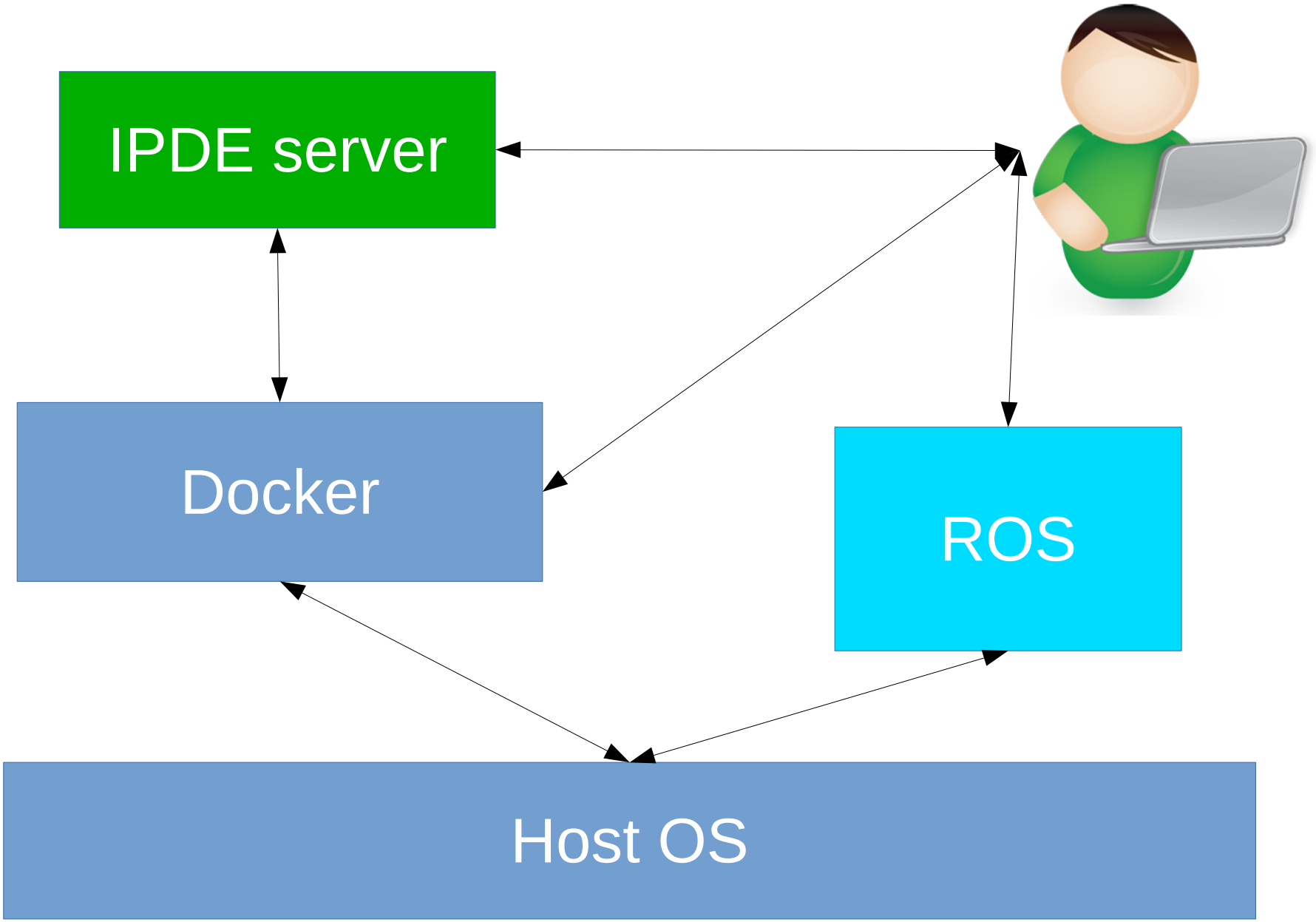
- Docker-based deployment system for ROS applications
- Deployed with Docker ;)
- Integrates software from different sources
- “Extended roslaunch”
  - Pulls in all dependencies
  - Builds source code
  - Keeps host clean



Software sources

- ROS packages:
  - Binary release
  - Source release
  - Locally developed
- Different stages:
  - set-up, testing, production, maintenance
- Different users:
  - developers, system integrators, end users

IPDE overview

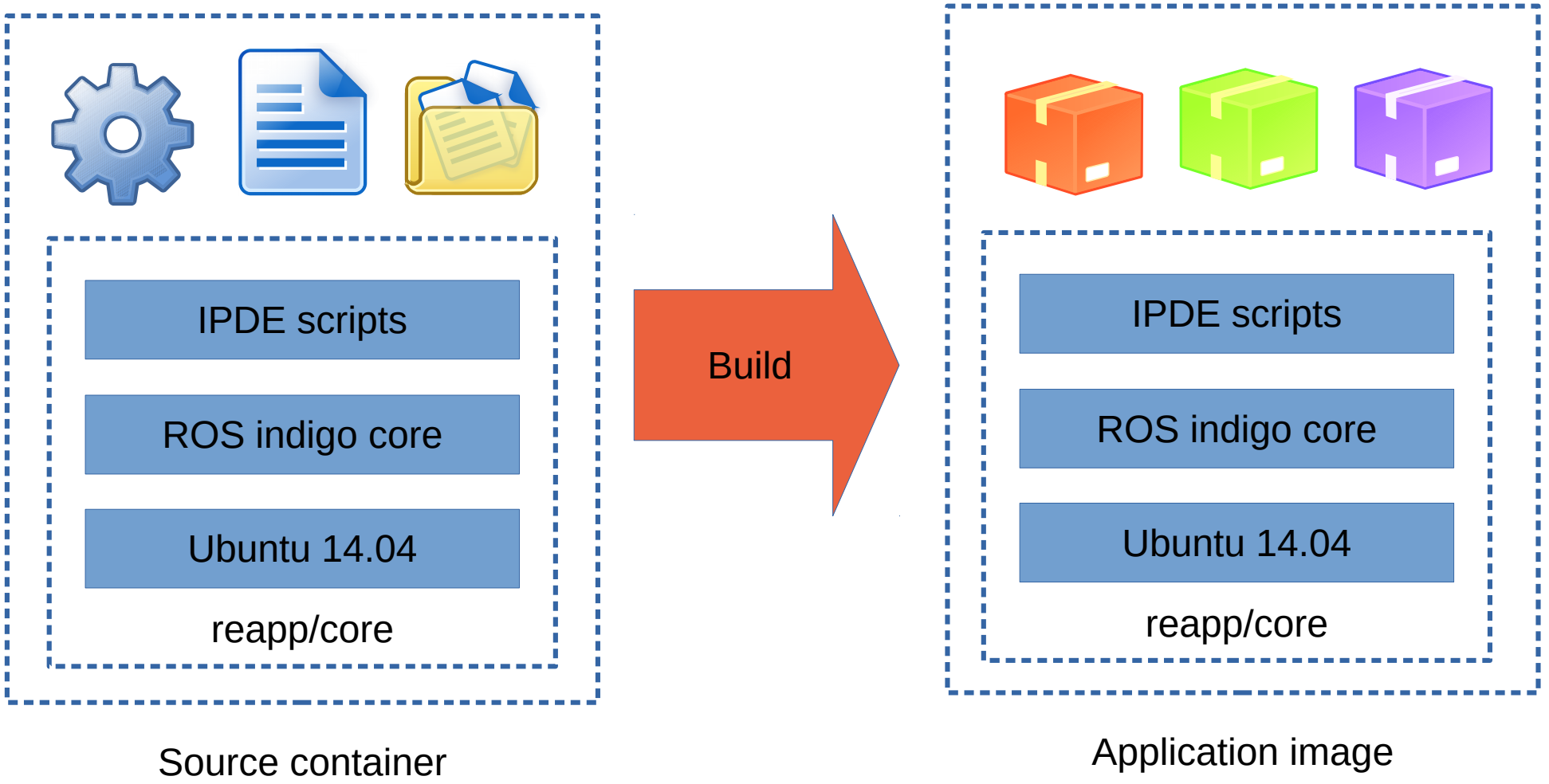


IPDE server

- Server written in Python with Flask
- REST API (XML and JSON)
- Handles all process logic and sessions
- Not involved at application run-time
- Deployed via Docker Hub
  - tiangolo/uwsgi-nginx-flask:flask :)
- Orchestrated with docker-compose
  - sameersbn/apt-cacher-ng :)

Session lifecycle

- “remote workspace” → application image



IPDE clients

- ReApp project
  - Eclipse-based Engineering Workbench
  - Web-based Solution Wizard
- Factory-in-a-Day
  - Shell utilities (→ Hands-on session)
  - Integration with the Automated Test Framework

Recap & Benefits

- Docker-based cross-platform deployment system for ROS applications
- Simplifies ROS source code workflow
- Turns ROS packages into isolated, self-contained application images
- Sessions allow separation of concerns
- Minimum overhead
- Server and clients freely available

Limitations

- X forwarding is difficult with Docker
- Building from scratch is rather slow
- No persistent local data storage
- Dependency management with rosdep only
- No user access levels
- Only one host is supported for now

Outlook / Nice-to-have

- Multi-host support for distributed application
- Automatic start-up of default applications
- A Web front-end would be great :)
- CLI utilities need polishing:
  - tab completion
  - single tool instead of different scripts

Questions?