An agile pathway towards carbon-aware clouds

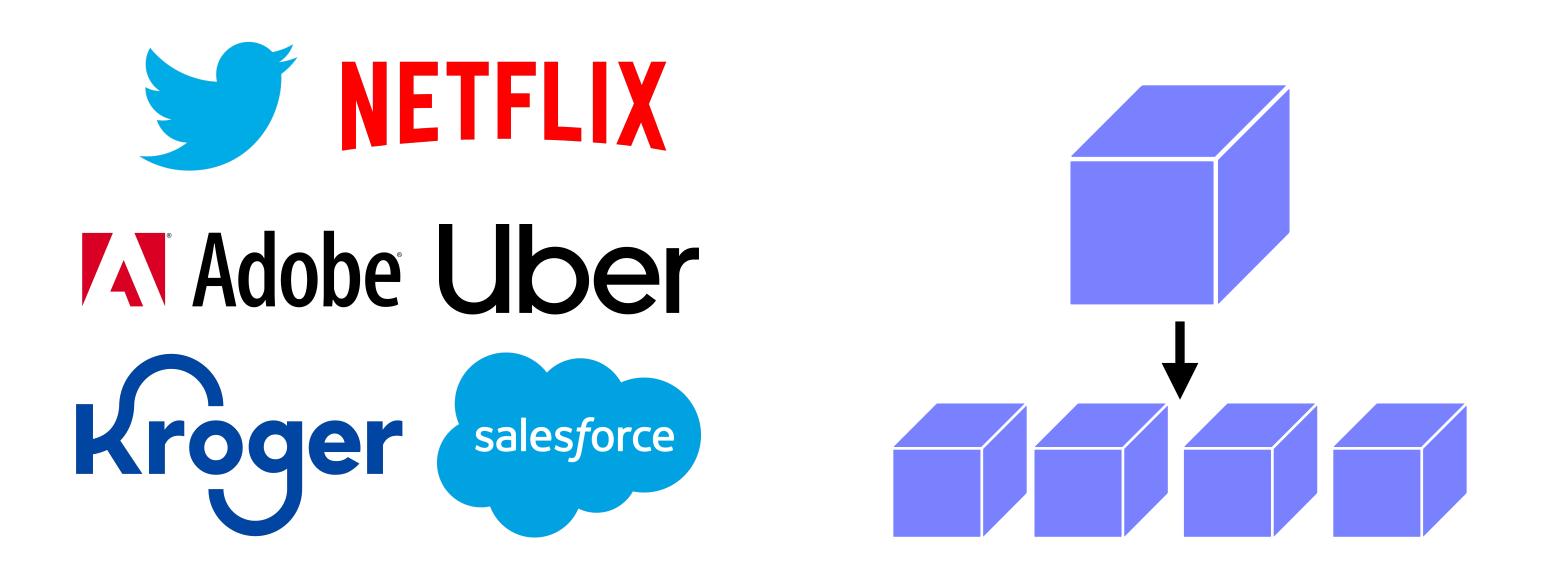
Pratyush Patel, Theo Gregersen, Tom Anderson

WUNIVERSITY of WASHINGTON

Disclaimer: an opinionated talk with systems implications — let's discuss!

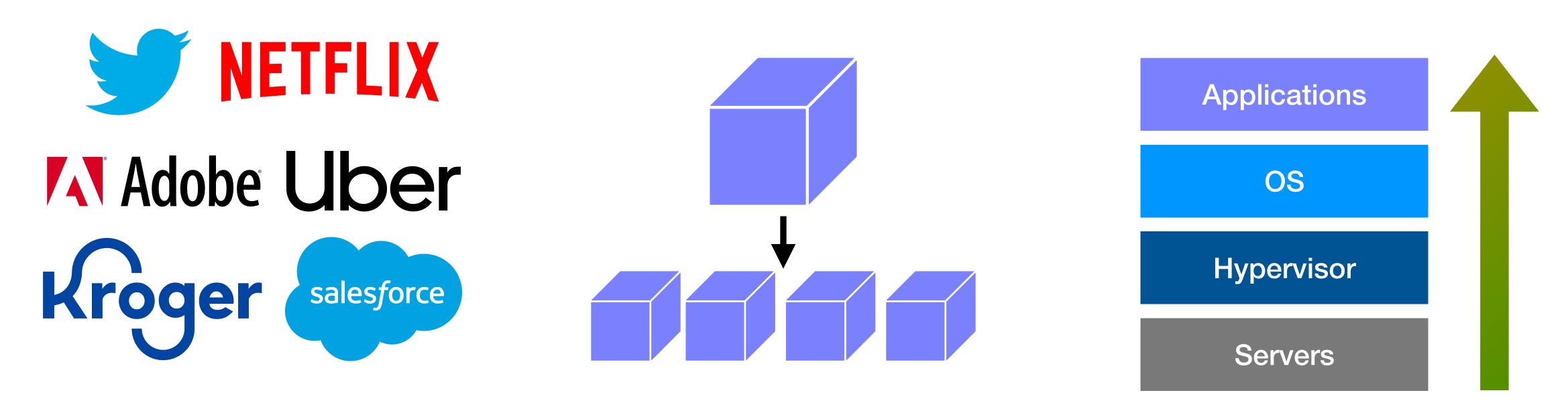


Cloud users are massive organizations



Cloud users are massive organizations

Applications span thousands of microservices

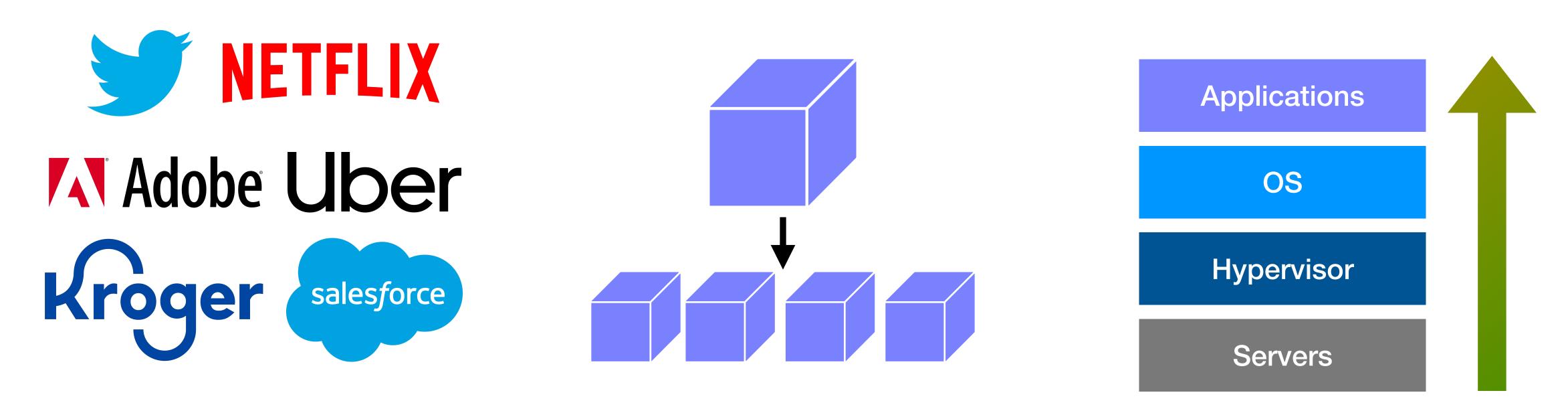


Cloud users are massive organizations

Applications span thousands of microservices

Carbon awareness must percolate the stack





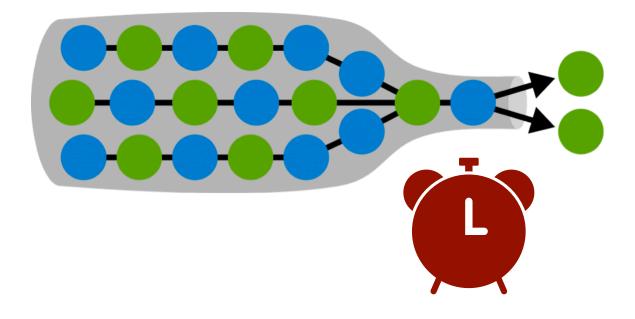
Cloud users are massive organizations

Applications span thousands of microservices

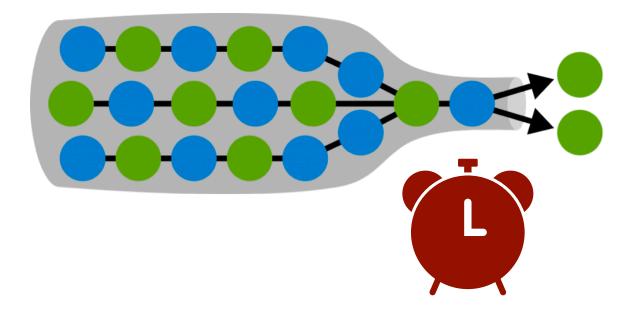
Carbon awareness must percolate the stack

Clearly, a very challenging problem!





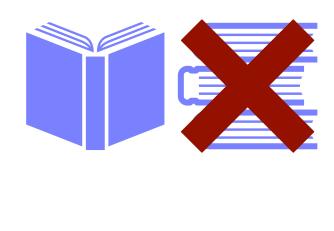
Developer time is a business bottleneck

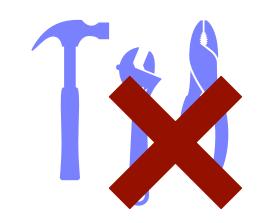


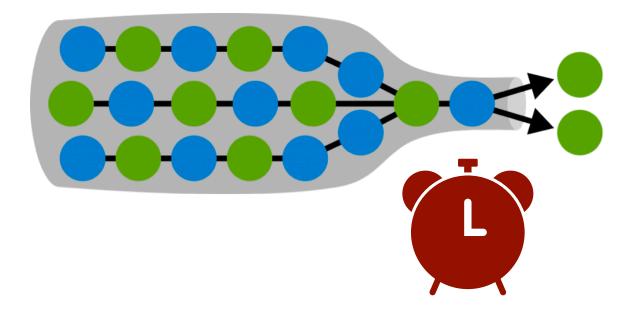


Developer time is a business bottleneck

Minimal training and tooling available



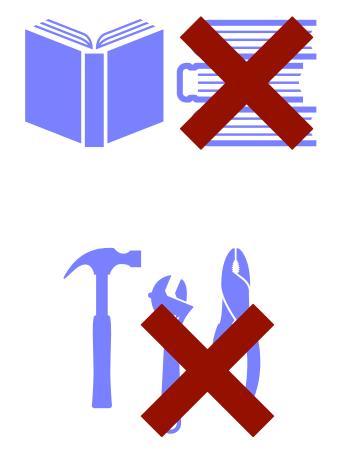


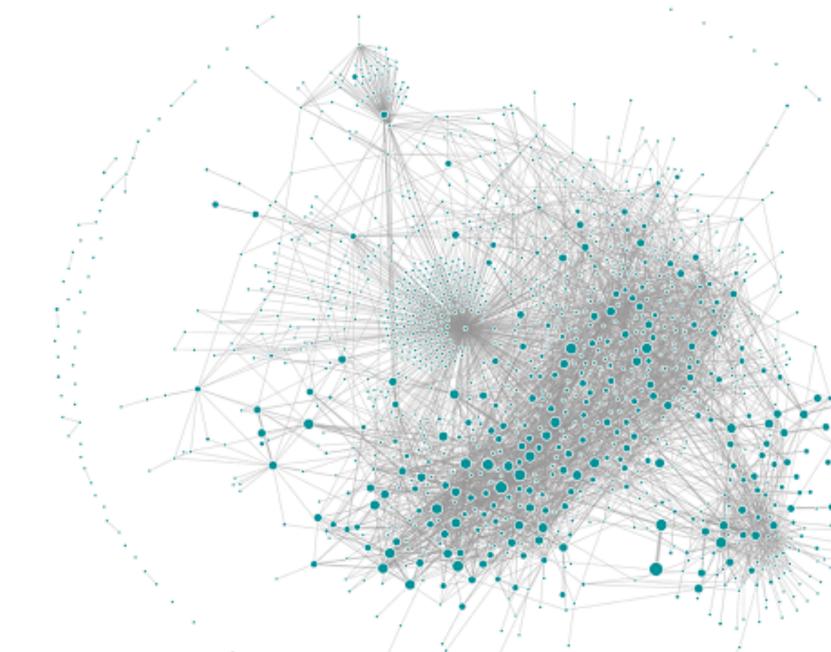




Developer time is a business bottleneck

Minimal training and tooling available





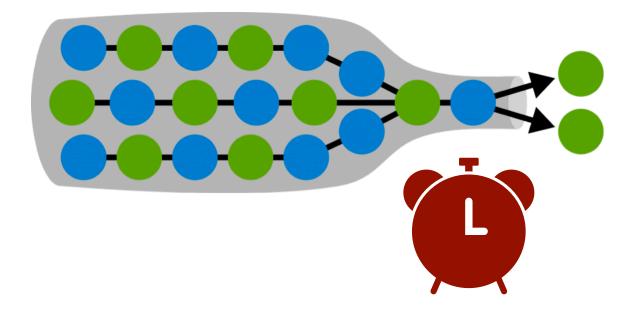
Rewriting apps may be infeasible

Uber's ~2,200 microservices visualized by Jaeger (2018)



•

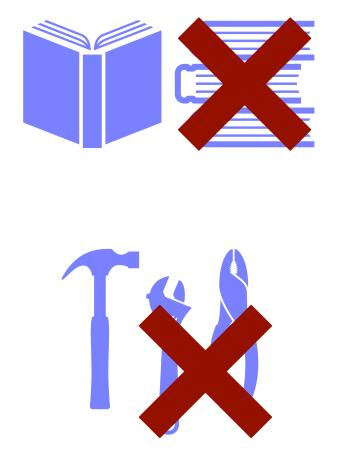
(2018)

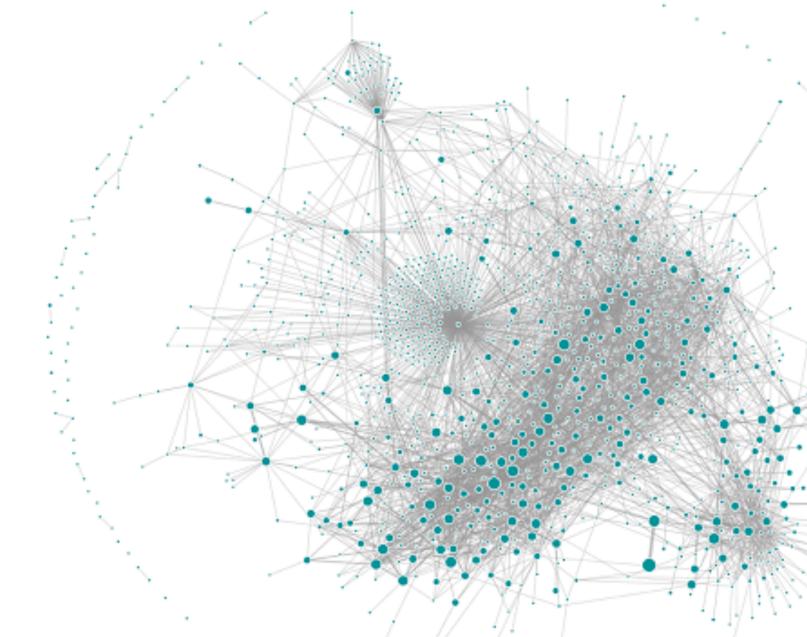




Developer time is a business bottleneck

Who can make an impactful and timely difference? How can we help them?





Minimal training and tooling available

Rewriting apps may be infeasible

Uber's ~2,200 microservices visualized by Jaeger (2018)





A closer look at large-scale application teams

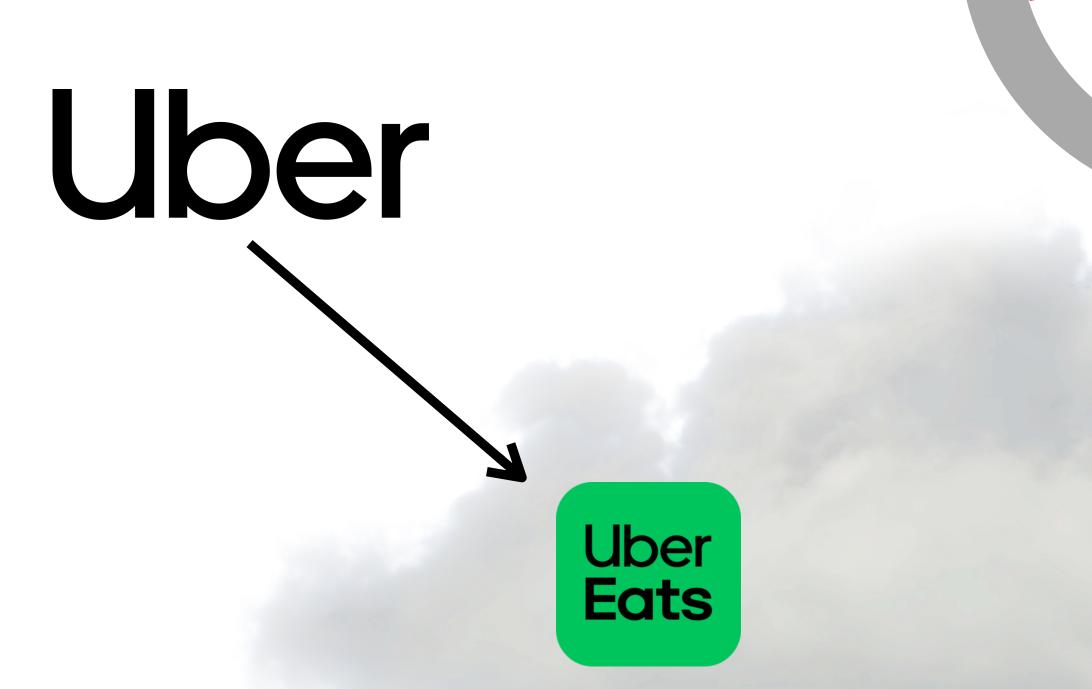






A closer look at large-scale application teams

NETFLIX



Adobe[®]

K

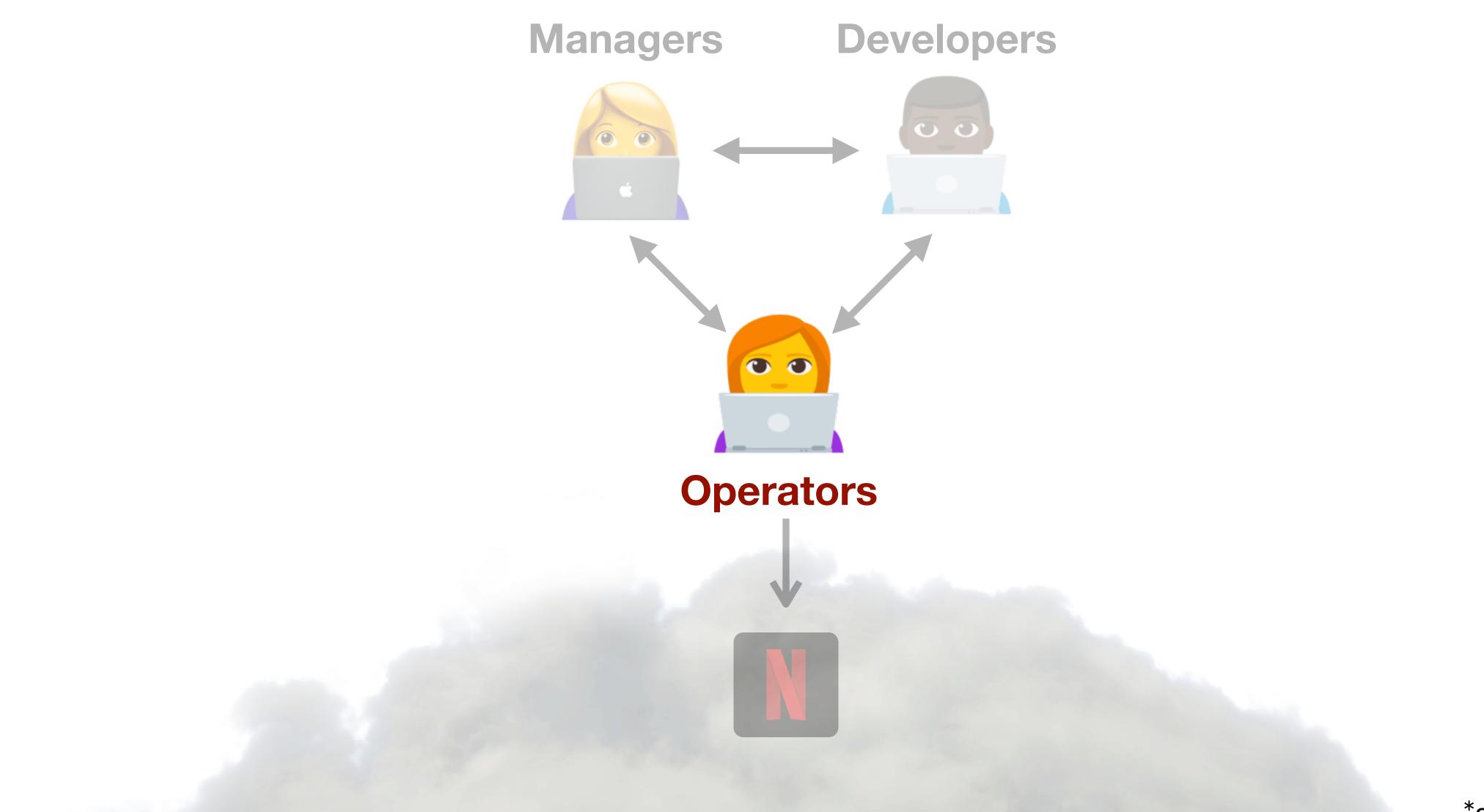
PS





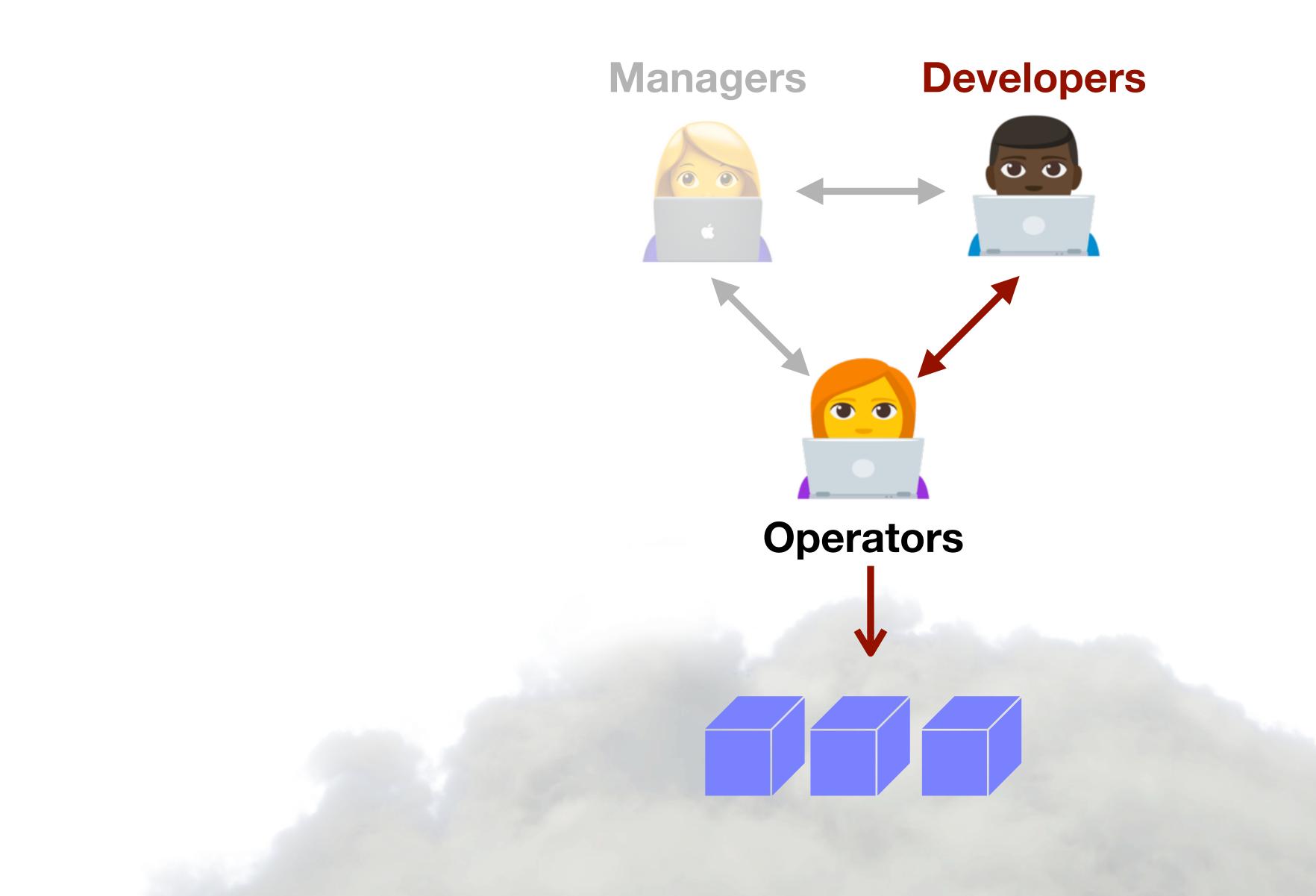


Every large-scale application team has an operations team





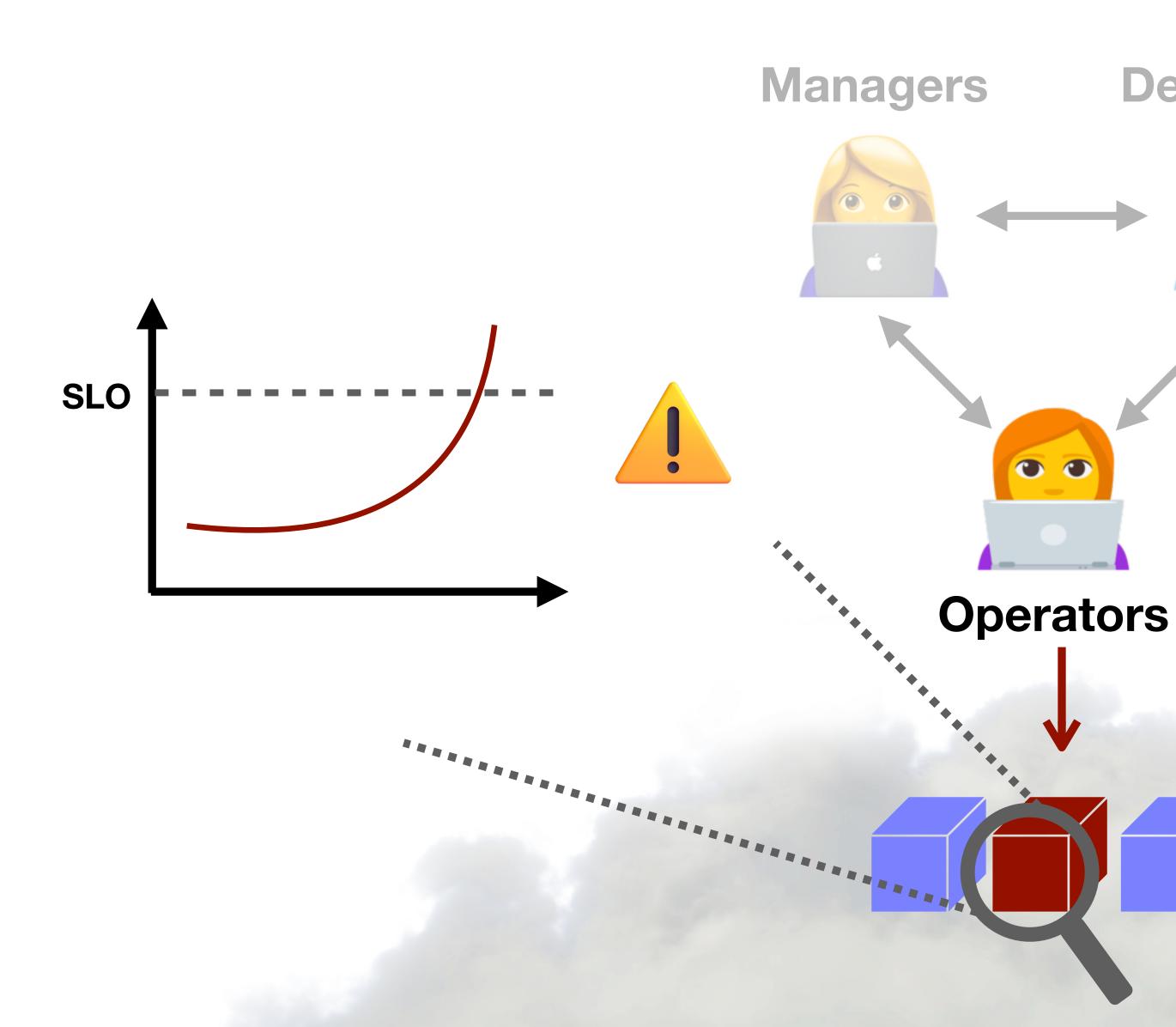
Operators work with developers to deploy applications on the cloud





Operators monitor deployments

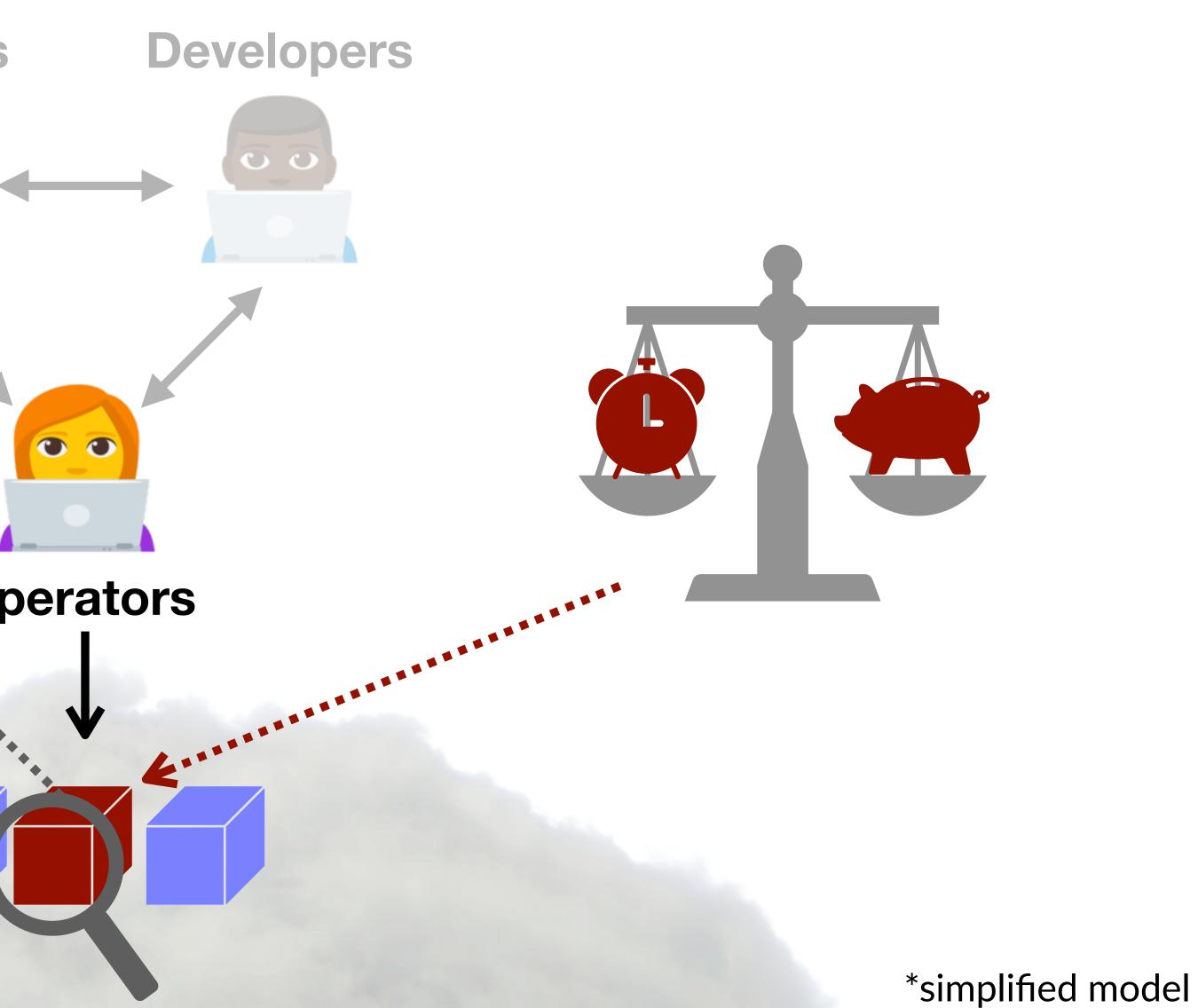
Developers





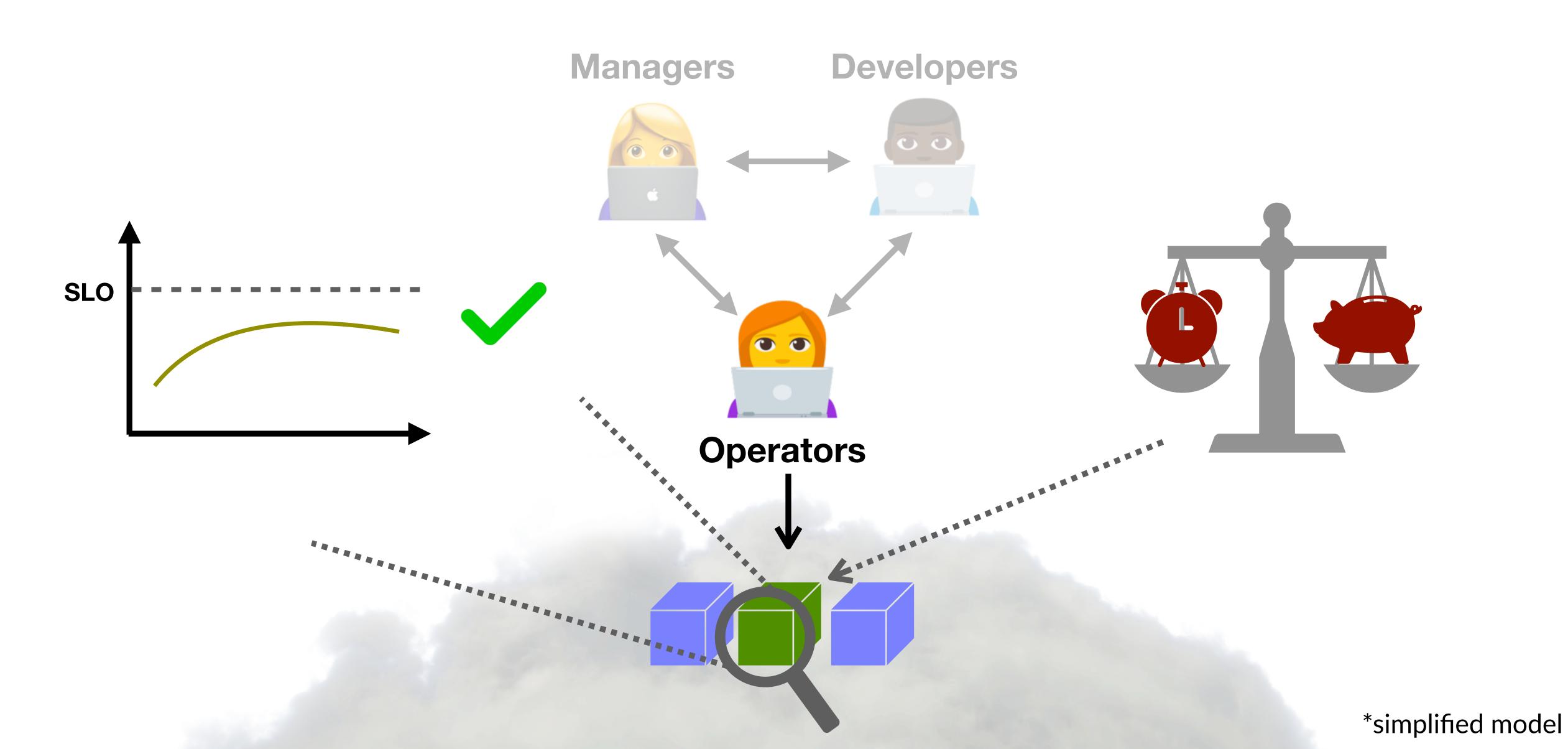


Operators monitor and make trade-offs Managers **Developers SLO** C 0 *** **Operators** ***** -----



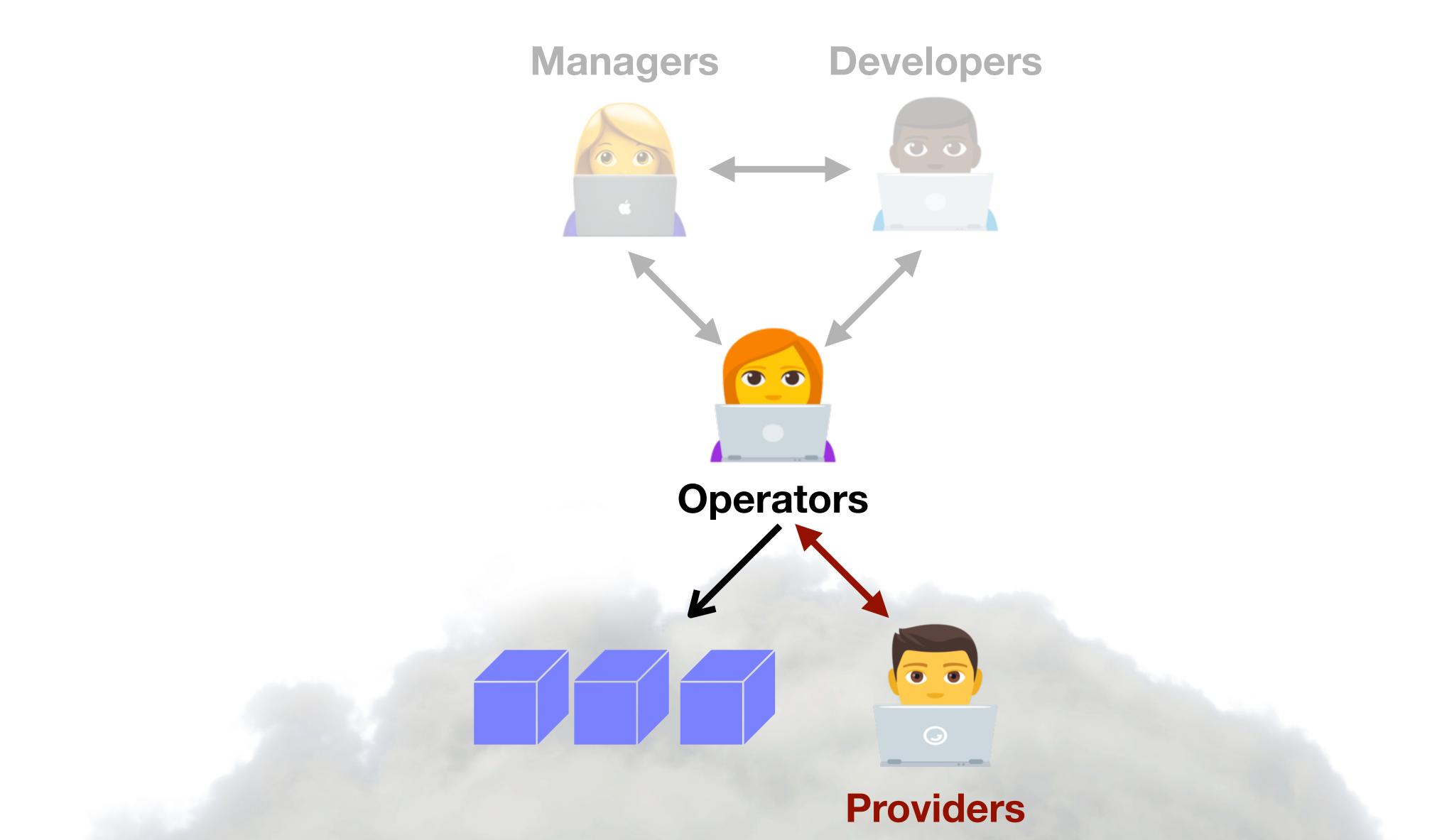


Operators monitor and make trade-offs to meet service objectives



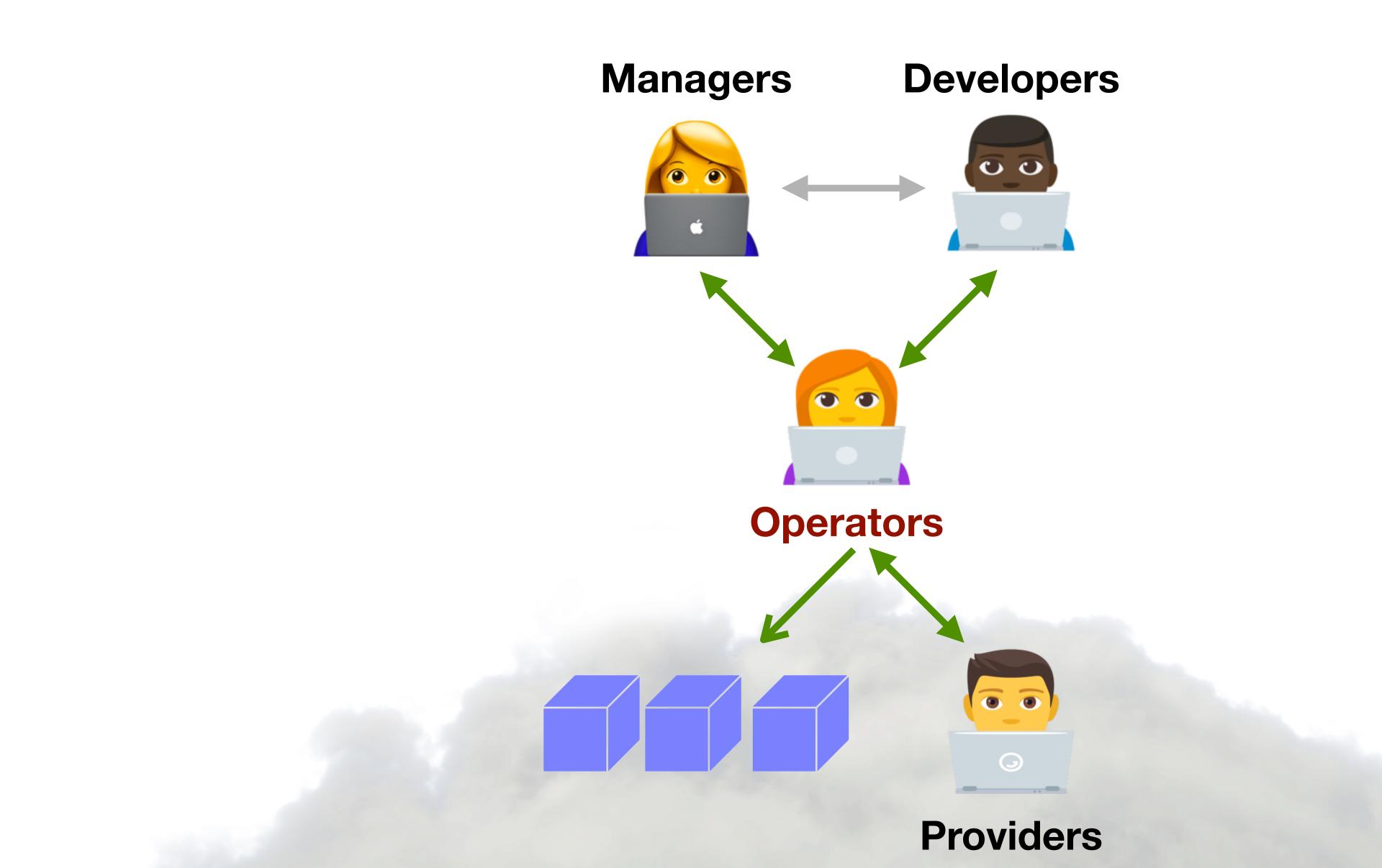


Operators also work closely with cloud provider teams





Operators are the narrow waist to enable cloud carbon awareness!



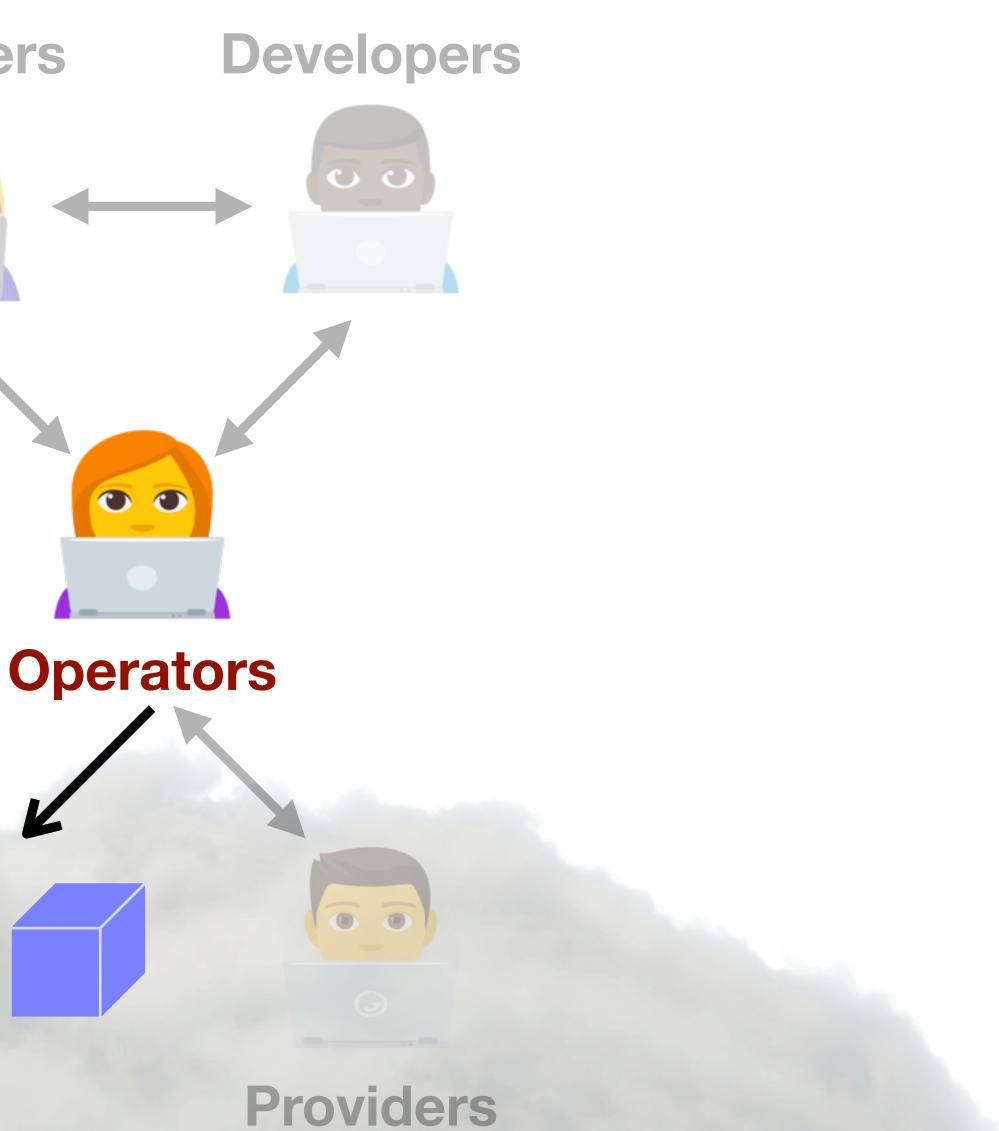


Viewing carbon as an operations concern

Managers

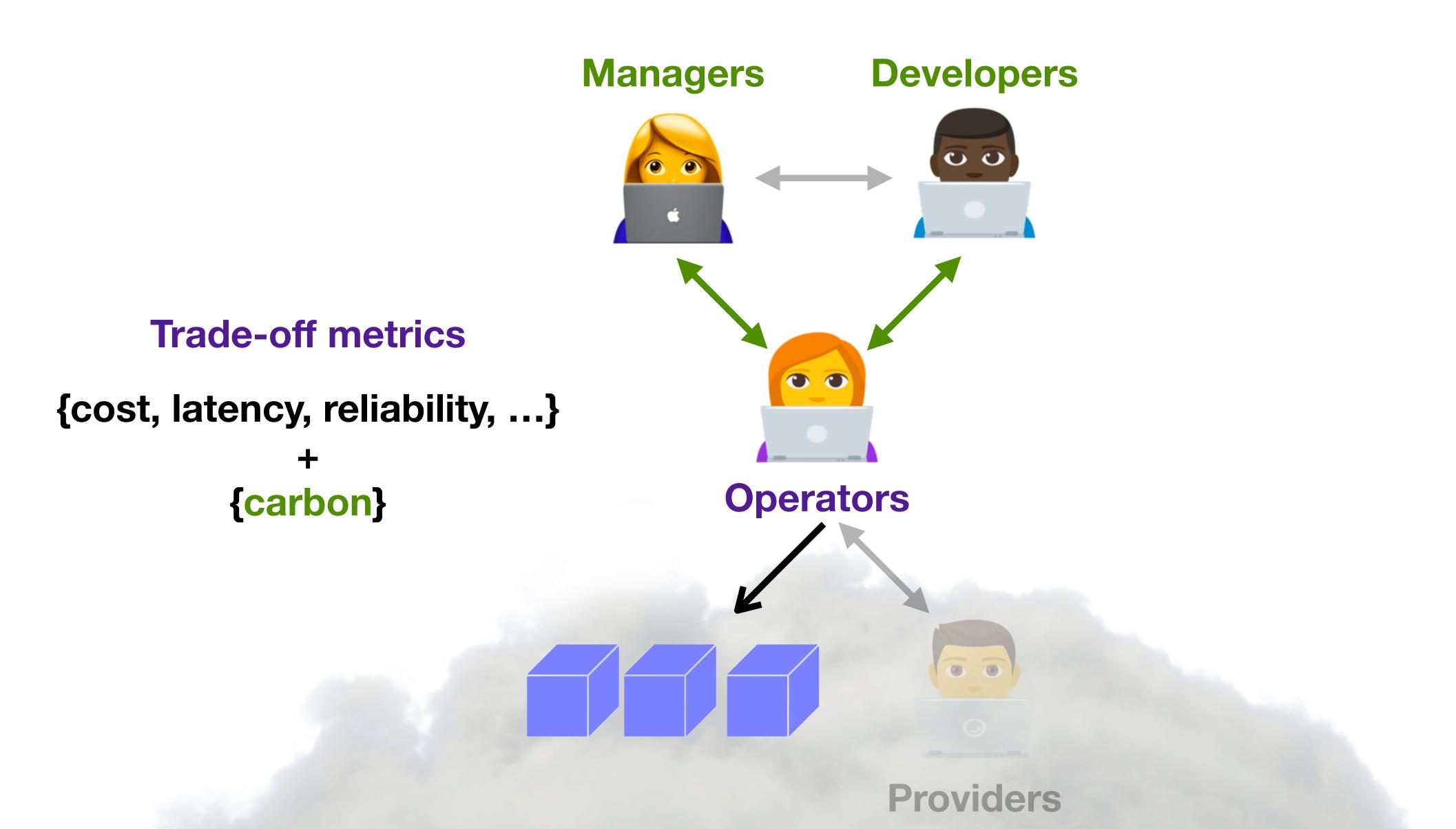
Trade-off metrics

{cost, latency, reliability, ...} ╋ {carbon}



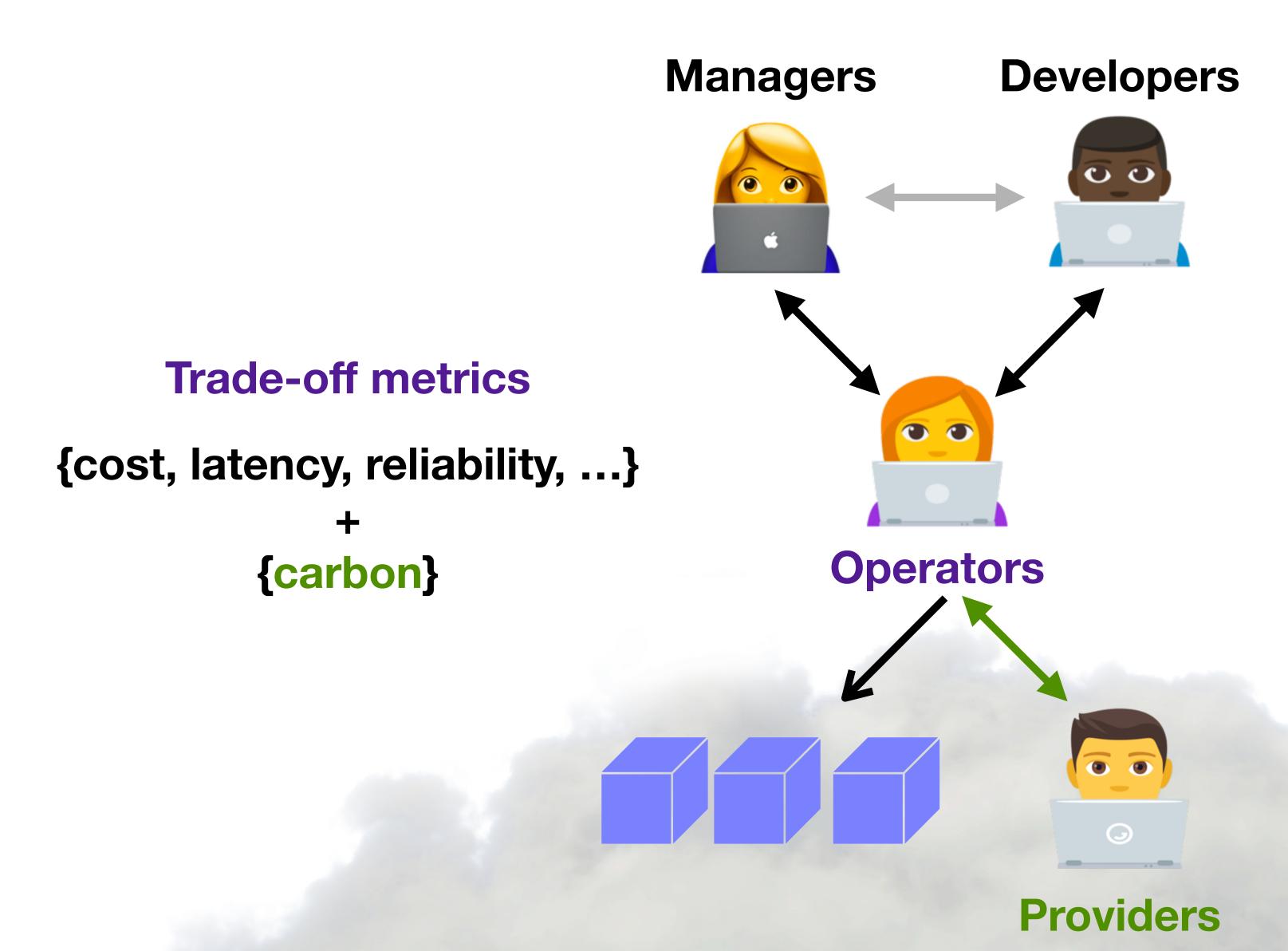


Operators can help managers and developers be carbon aware





Operators and providers can communicate visibility to reduce carbon



Visibility is needed in both directions!

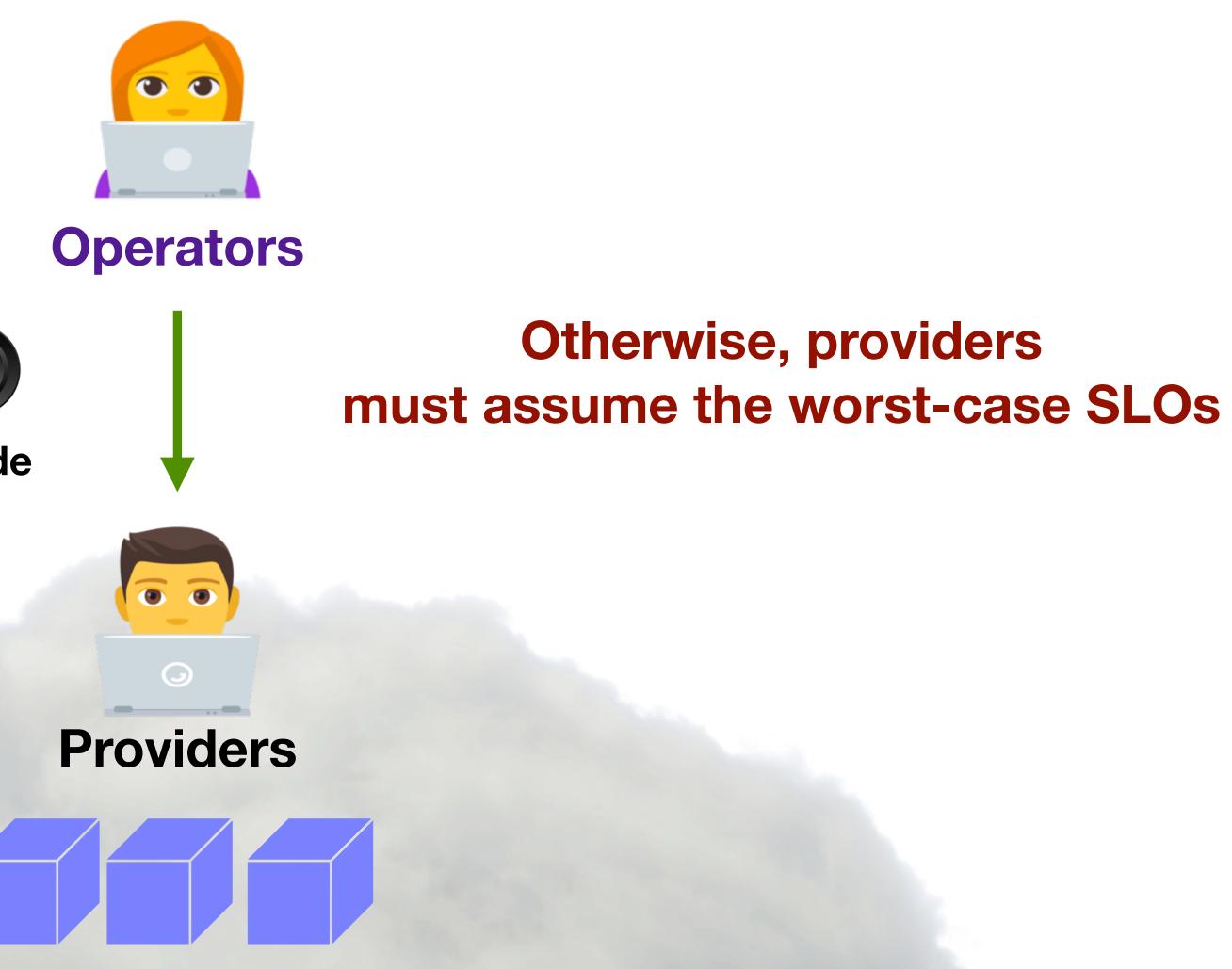


Operator—>provider communication with application "eco modes"

capture coarse-grained notions of carbon trade-offs

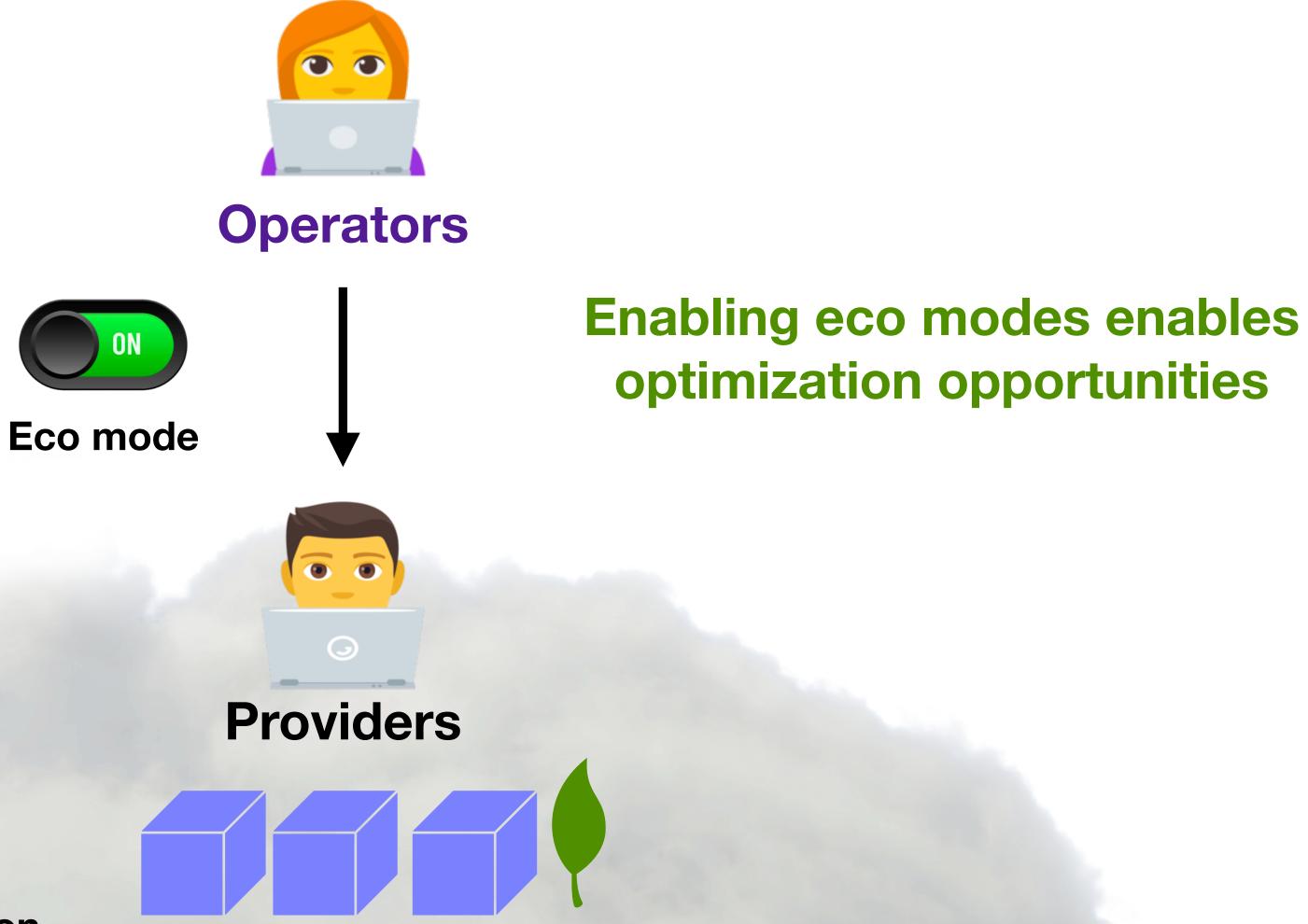






Toggle applications between power mode and eco mode

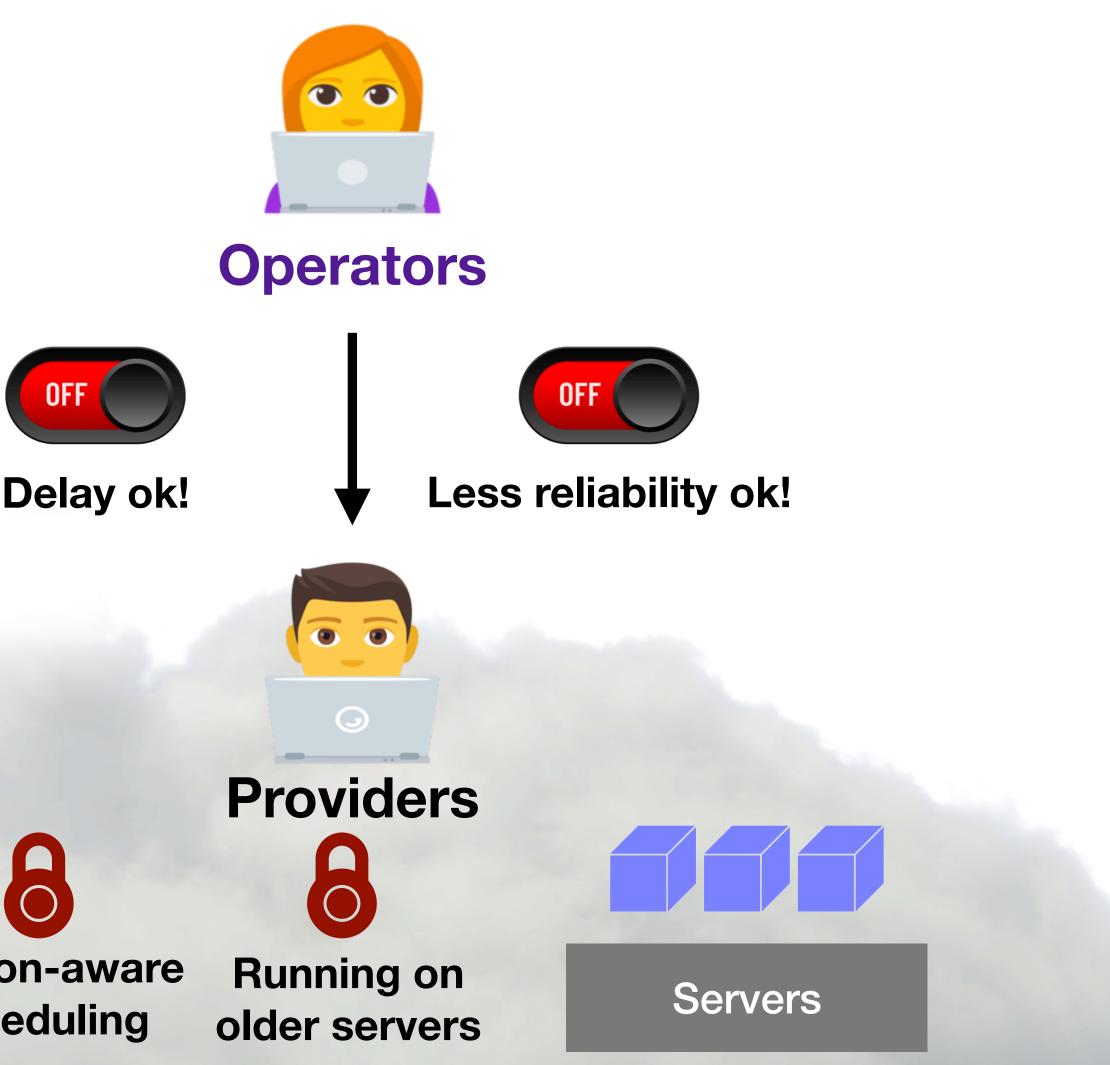
capture coarse-grained notions of carbon trade-offs

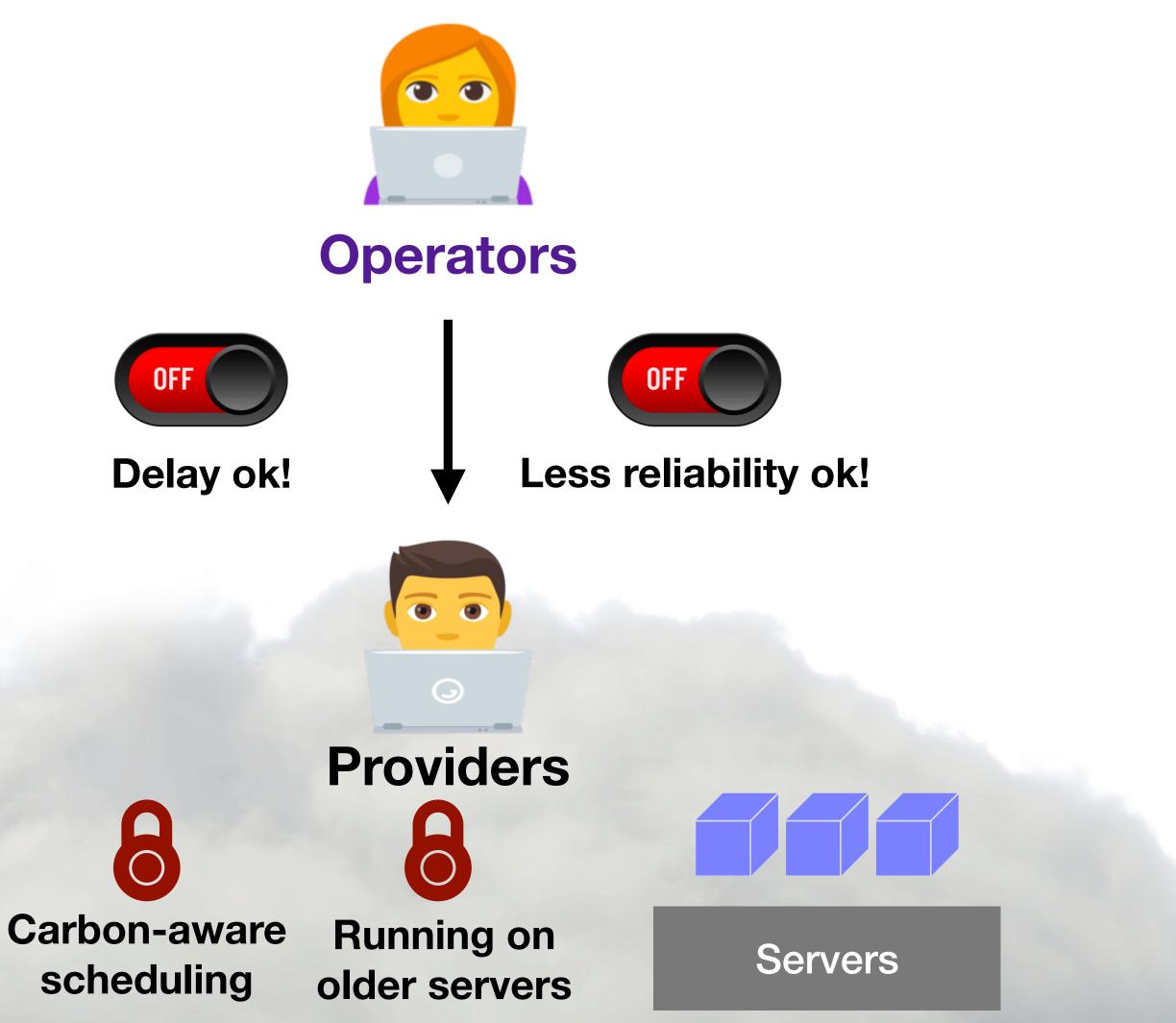




Different eco modes, different trade-offs

capture coarse-grained notions of carbon trade-offs

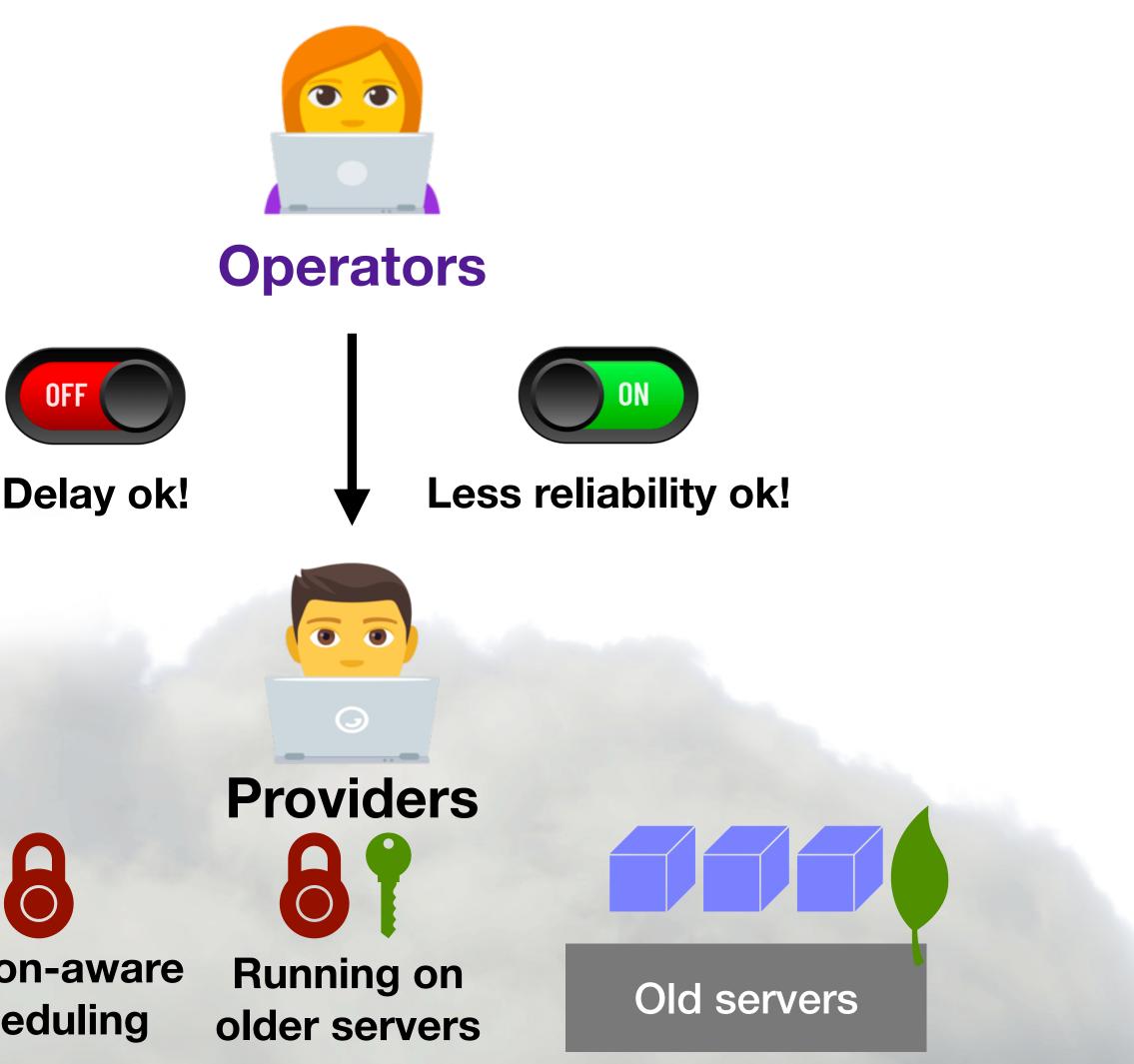


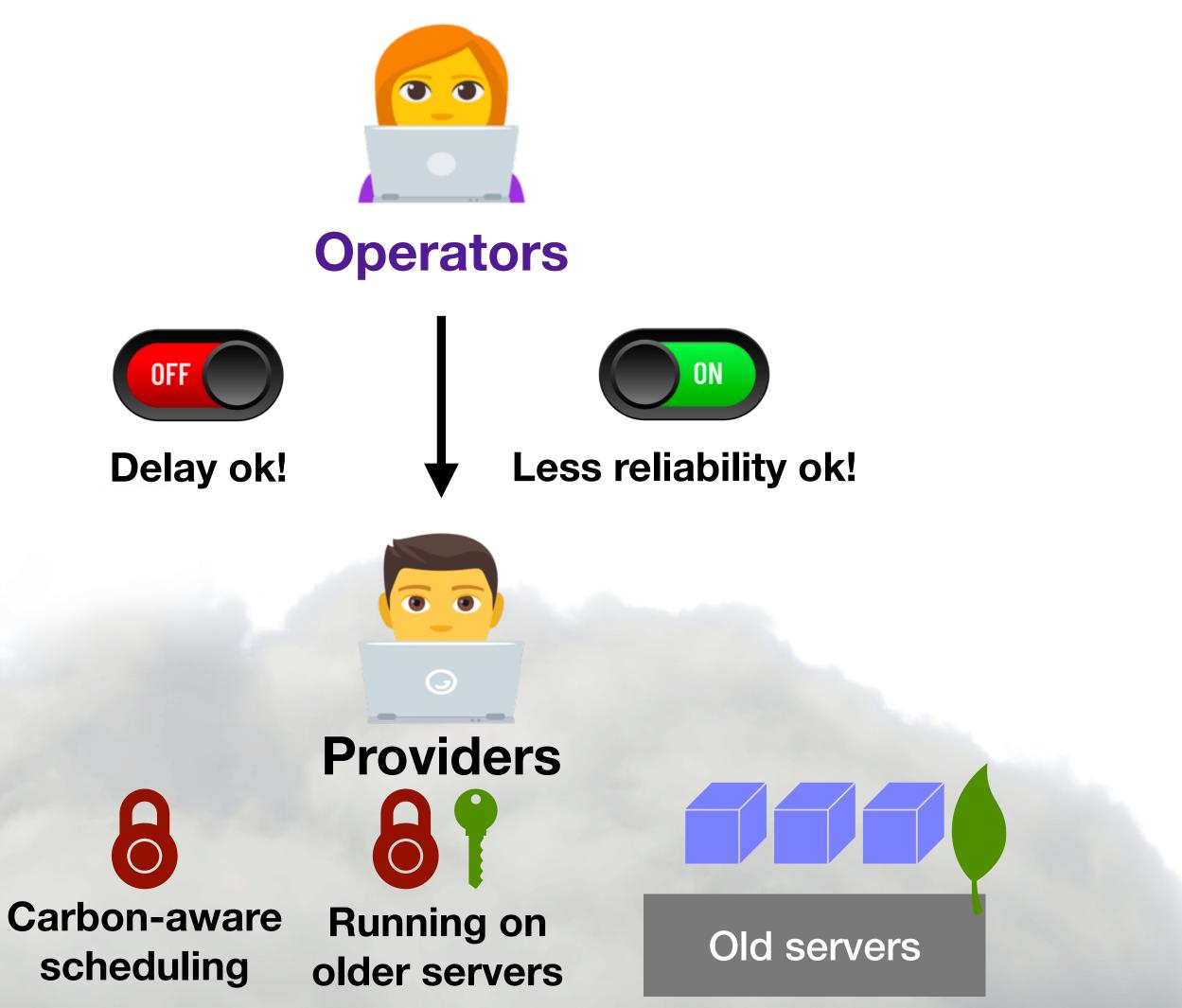


Different eco modes, different trade-offs

capture coarse-grained notions of carbon trade-offs

. . .





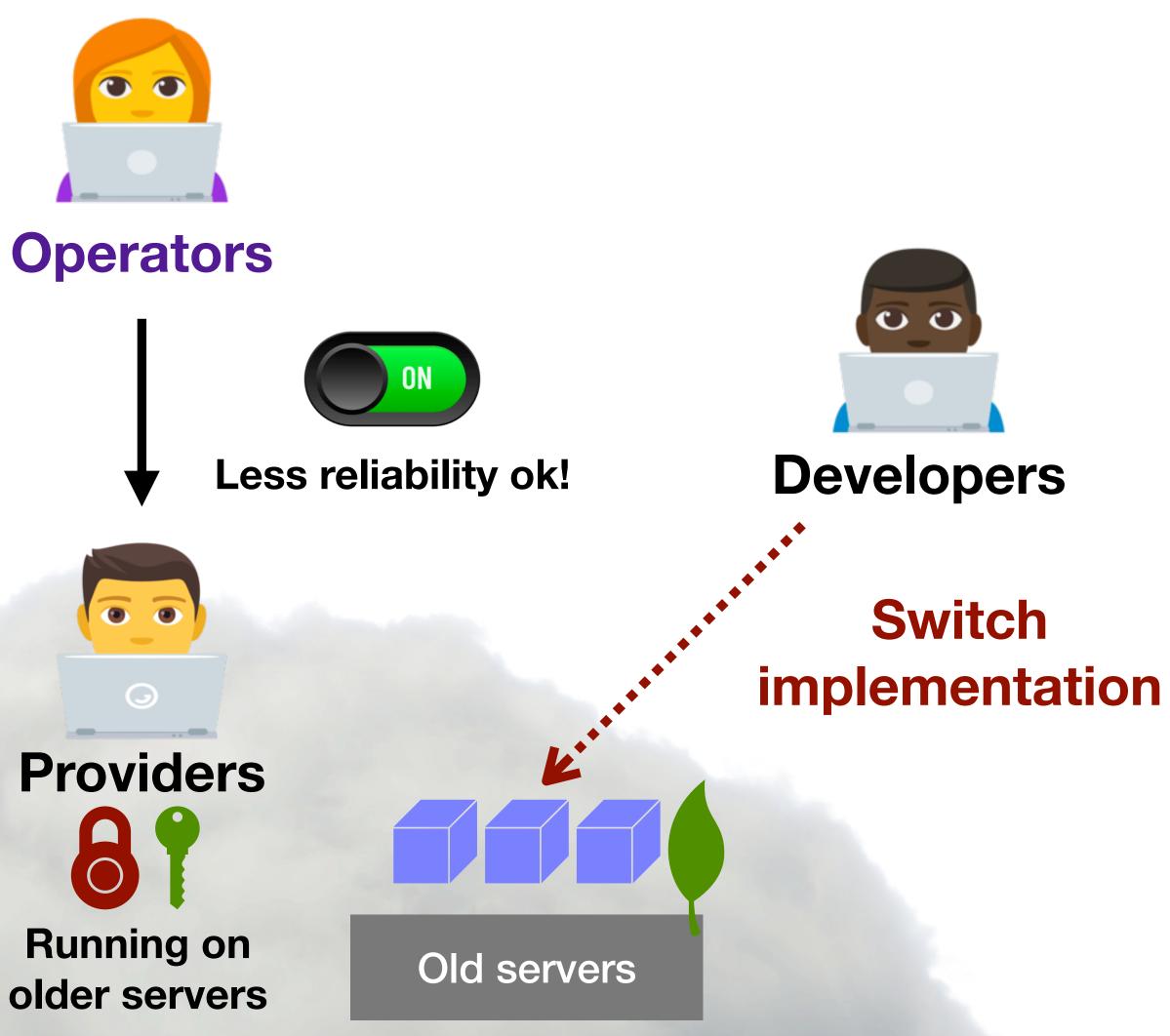
Developers can extend eco modes by implementing new trade-offs

capture coarse-grained notions of carbon trade-offs





. . .



Provider—>operator communication with service-level carbon metrics

empower application operators with carbon visibility

00

Operators



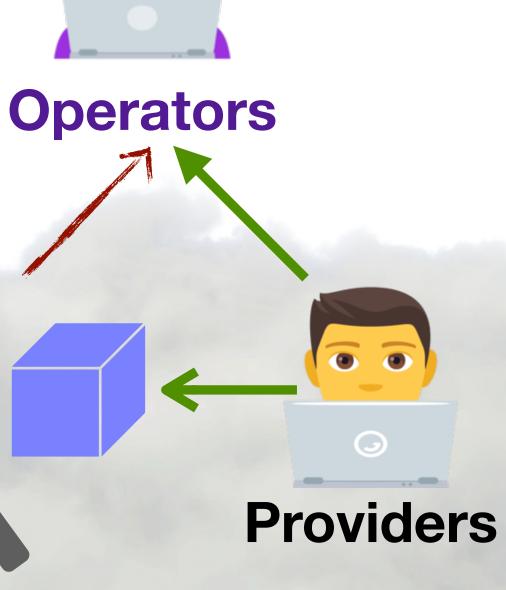


Approximate metrics are good enough! (for now)

empower application operators with carbon visibility

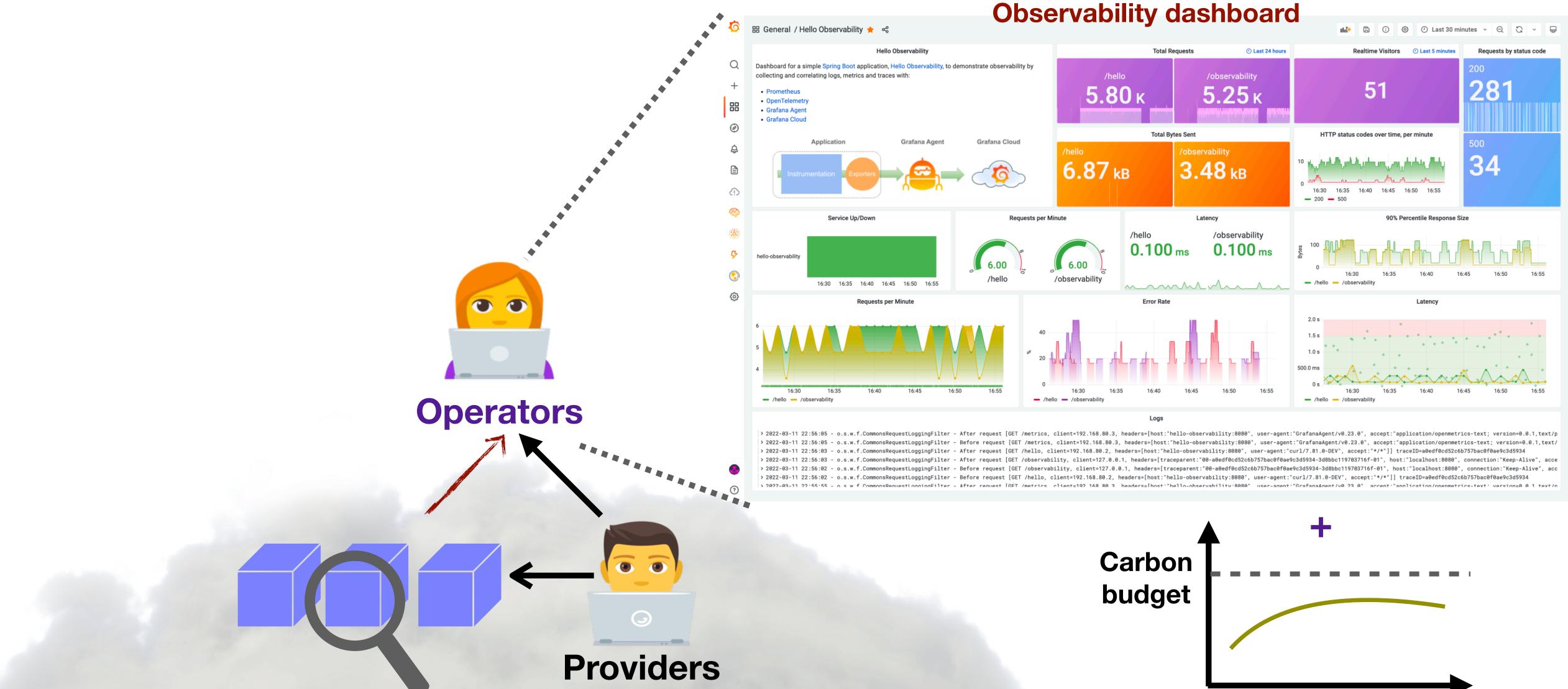
 \odot \odot



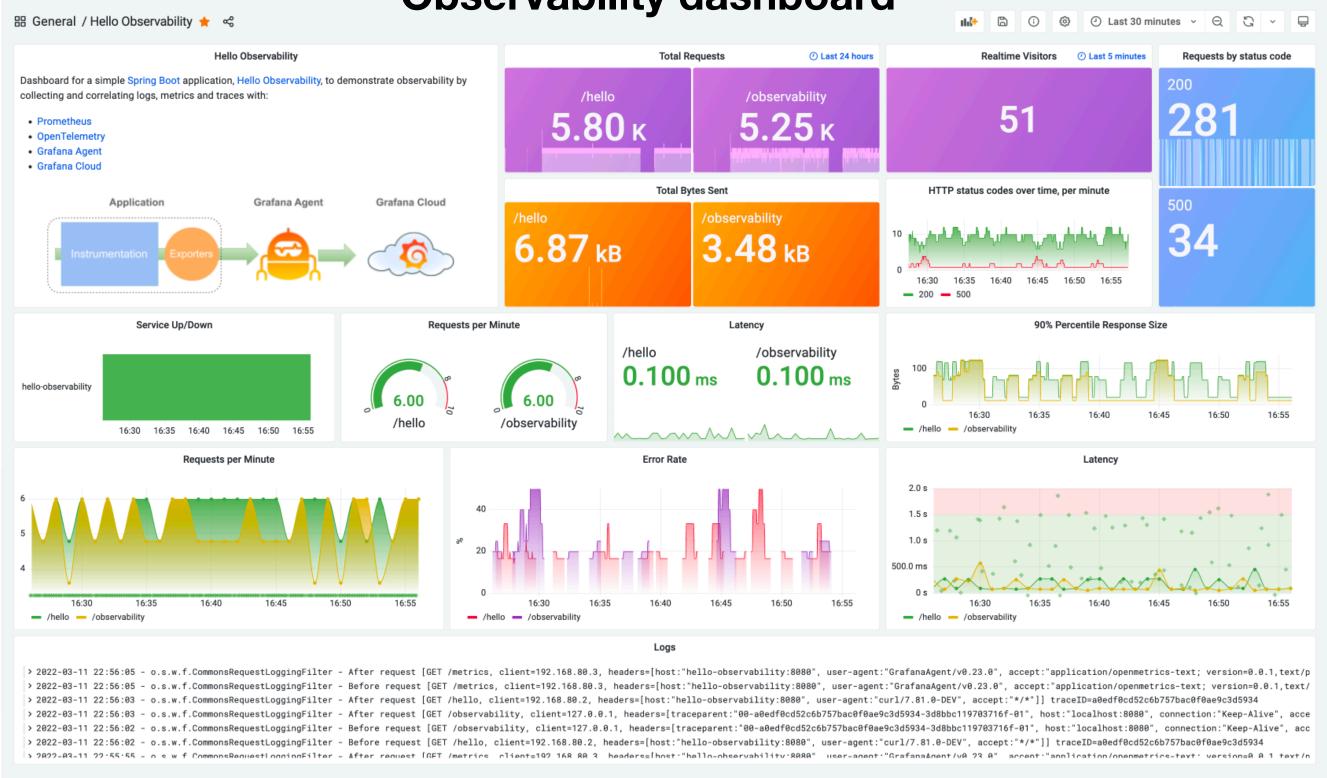


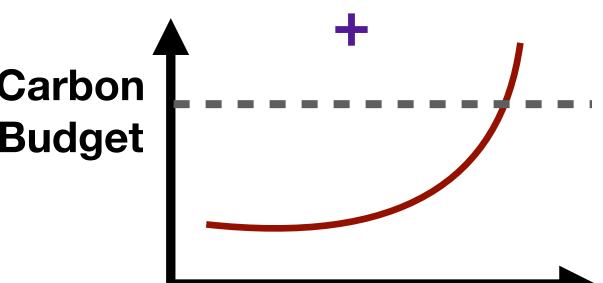
Integrate carbon into existing operator workflows

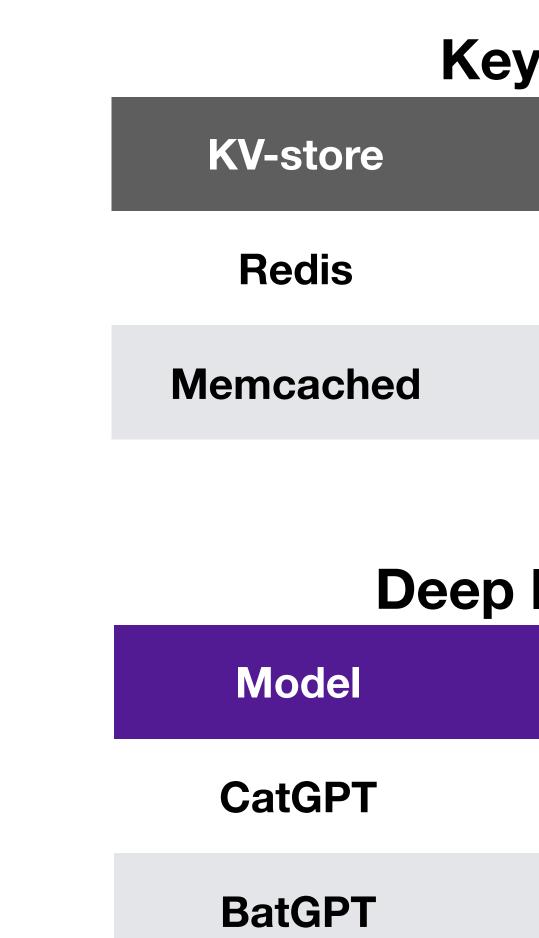
empower application operators with carbon visibility



Helps identify carbon hotspots for developers to optimize empower application operators with carbon visibility **Observability dashboard** 🔠 General / Hello Observability 🔶 😪 ① Last 24 hours Total Request Realtime Visitors ② Last 5 minutes Hello Observabilit Ο Dashboard for a simple Spring Boot application, Hello Observability, to demonstrate observability by /observability collecting and correlating logs, metrics and traces with: /hello 51 5.80 k 5.25 k Prometheu OpenTelemetr 88 Grafana Agent Grafana Cloud Ø Total Bytes Sent HTTP status codes over time, per minute Grafana Agen Grafana Cloud /observability المراجعة والماري المارية 6.87 kB 3.48 kB 16:30 16:35 16:40 16:45 16:50 16:55 \odot - 200 - 500 Latency Service Up/Dowr Requests per Minute /hello /observability 0.100 ms **0.100** ms Г, hello-observability 6.00 6.00 /hello /observability 16:30 16:35 16:40 16:45 16:50 16:55 -----00 Requests per Minute Error Rate Latency \odot 2.0 s 1.5 s 1.0 s 16:50 16:30 16:35 16:45 **Developers Operators** > 2022-03-11 22:56:05 o.s.w.f.CommonsRequestLoggingFilter - After request [GET /metrics, client=192.168.80.3, headers=[host:"hello-observability:8080", user-agent:"GrafanaAgent/v0.23.0", accept: - Before request [GET /metrics, client=192.168.80.3, headers=[host:"hello-observability:8080", user-agent:"GrafanaAgent/v0.23.0", accept ***** sRequestLoggingFilter After request [GET /observability. client=127.0.0.1, headers=[traceparent:"00-a0edf0cd52c6b757bac0f0ae9c3d5934-3d8bbc119703716f-01 o.s.w.f.CommonsRequestLoggingFilter 2022-03-11 22:56:02 - o.s.w.f.CommonsReguestLoggingFilter - Before reguest [GET /observability, client=127.0.0.1, headers=[traceparent:"00-a0edf0cd52c6b757bac0f0ae9c3d5934-3d8bbc119703716f-01". request [GET /hello, client=192.168.80.2, headers=[host:"hello-observability:8080", o.s.w.f.CommonsRequestLoggingFilter user-agent:"curl/7.81.0-DEV" Carbon Budget **Providers**







"menu cards" for software carbon intensity

Key-value stores

Reads	Writes
10k req/s	5k req/s
12k/s req/s	4k req/s

Deep learning models

Accuracy	Latency
99%	64 ms/token
90%	48 ms/token



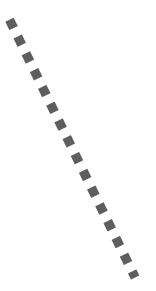
Key-value stores

KV-store	Reads	Writes	Carbon	
Redis	10k req/s	5k req/s	??	-
Memcached	12k/s req/s	4k req/s	??	

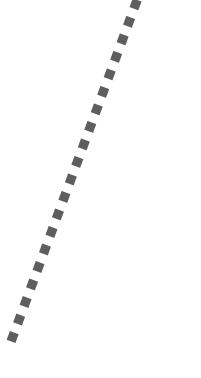
Deep learning models

Model	Accuracy	Latency	Carbon
CatGPT	99%	64 ms/token	??
BatGPT	90%	48 ms/token	??

"menu cards" for software carbon intensity

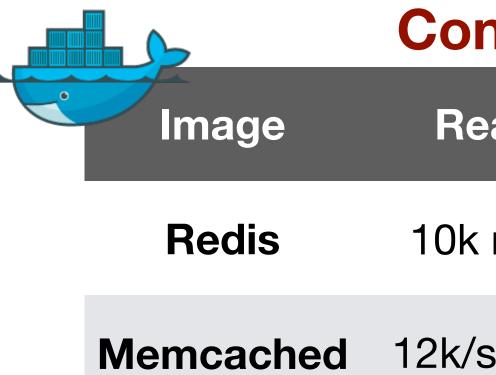


Standardize carbon benchmarks

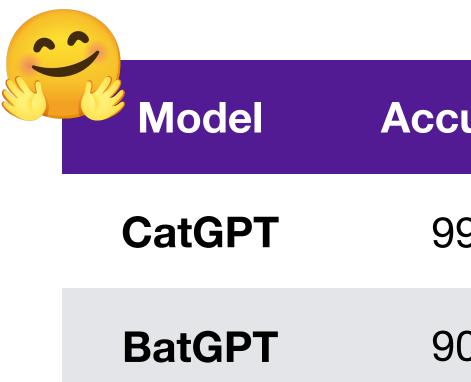








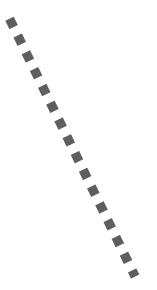
Integrate into popular repositories



"menu cards" for software carbon intensity

Container registry

eads	Writes	Carbon	
req/s	5k req/s	??	
s req/s	4k req/s	??	



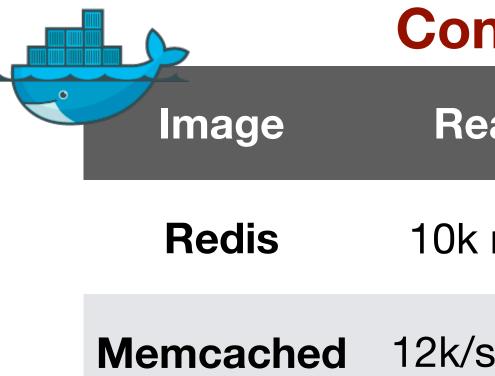
Standardize carbon benchmarks

Model zoo

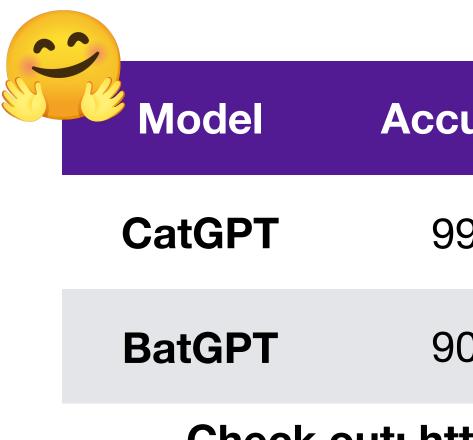
uracy	Latency	Carbon
9%	64 ms/token	??
0%	48 ms/token	??







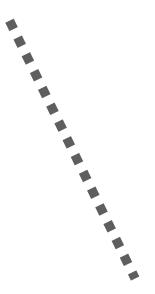
Integrate into popular repositories



"menu cards" for software carbon intensity

Container registry

eads	Writes	Carbon	
req/s	5k req/s	??	
s req/s	4k req/s	??	

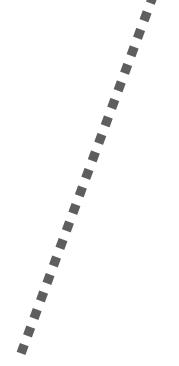


Standardize carbon benchmarks

Model zoo

uracy	Latency	Carbon
9%	64 ms/token	??
0%	48 ms/token	??

Check out: https://ml.energy/leaderboard/







An agile pathway towards carbon-aware clouds

1. Federate carbon responsibility and tools with operators as the narrow waist

2. Provide actionable visibility into carbon emissions

- 3. Centralize configurable optimizations for SLO-aware carbon reduction

Thanks! pratyush@cs.uw.edu

