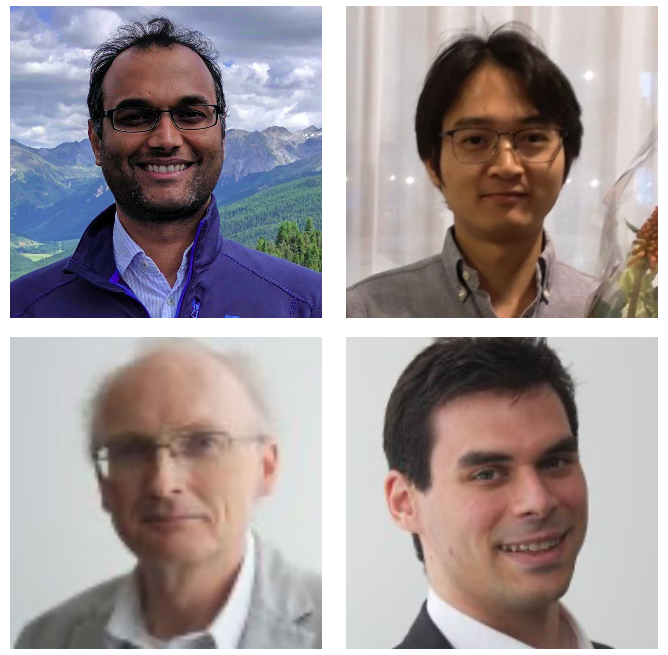
Sharing and Caring of Data at the Edge



Animesh Trivedi*, **Lin Wang***,
Henri Bal, Alexandru Iosup

USENIX HotEdge'20
June 25, 2020

VU VRIJE UNIVERSITEIT AMSTERDAM

TECHNISCHE UNIVERSITÄT DARMSTADT

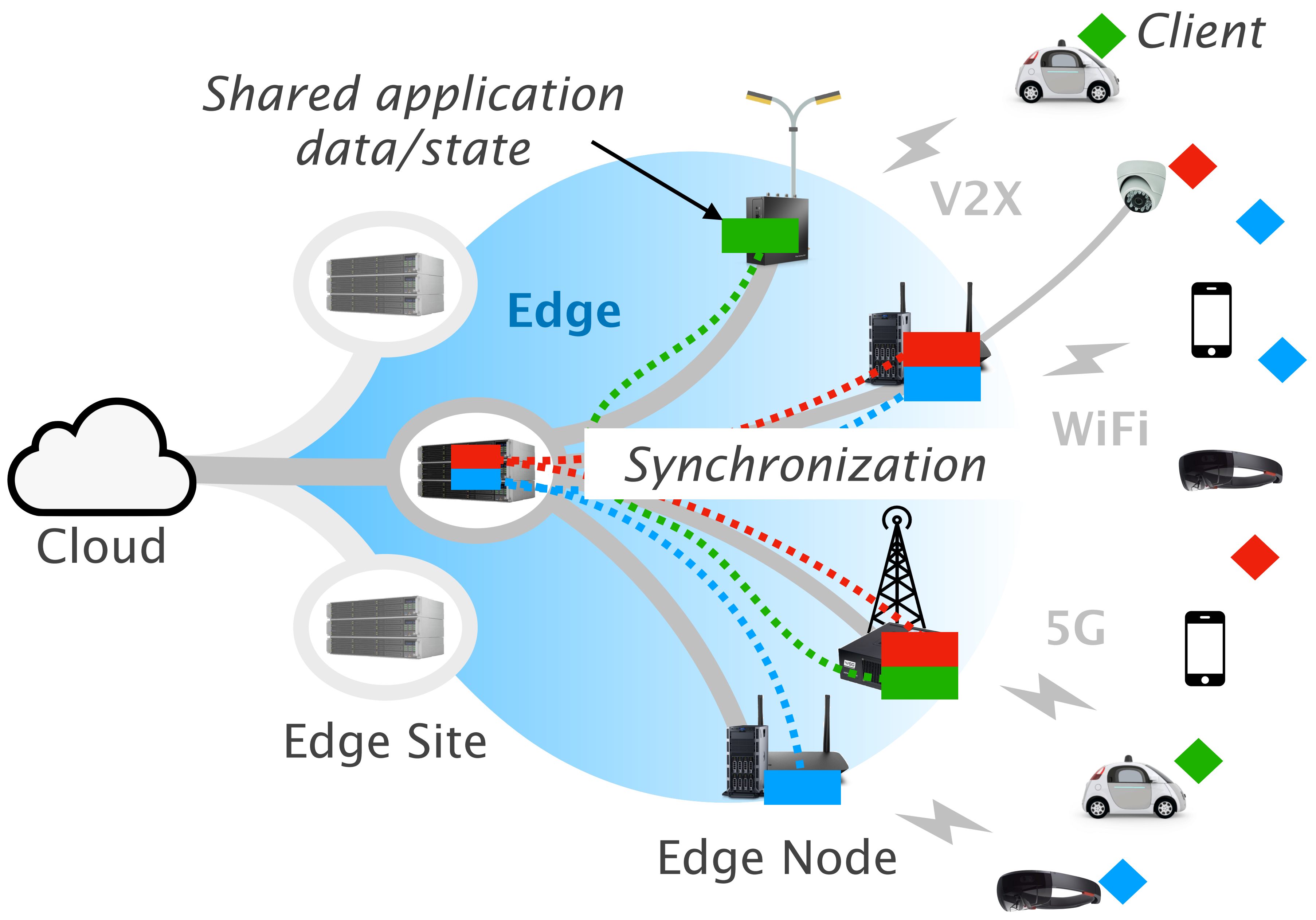
What do we mean when we talk about edge computing?

Edge infrastructure

- Distributed: cloud-edge continuum
- Heterogeneous: servers, Jetson boards, Raspberry Pis
- Dynamic: user mobility, resource reclamation

Edge applications: many are *collaborative*

- AR/VR/MR gaming: user profile, game state
- Autonomous driving: maps, LiDAR data, models
- IoT sensing/analytics: environment, tracking state
- Edge ML/DL: shared models/parameters, training data

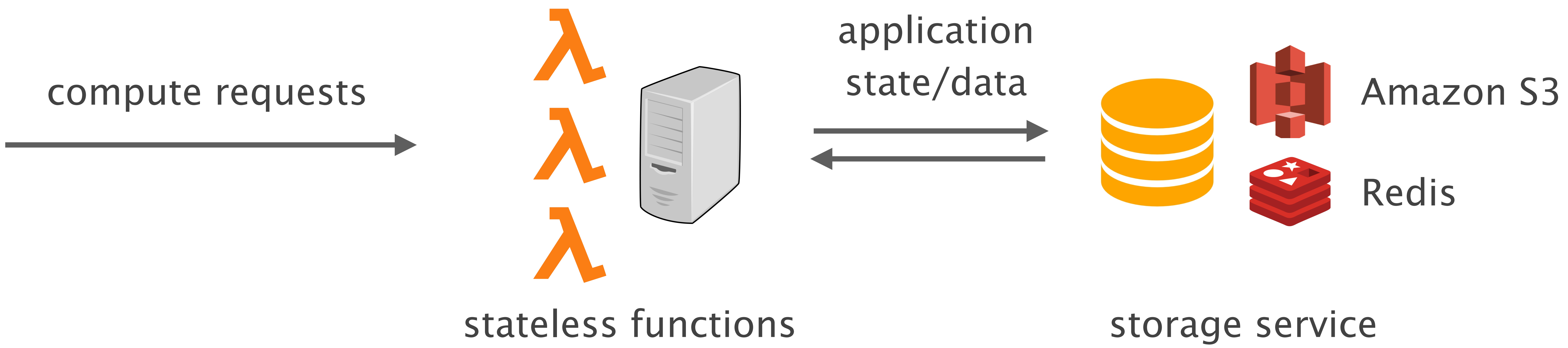


How do applications share data nowadays?

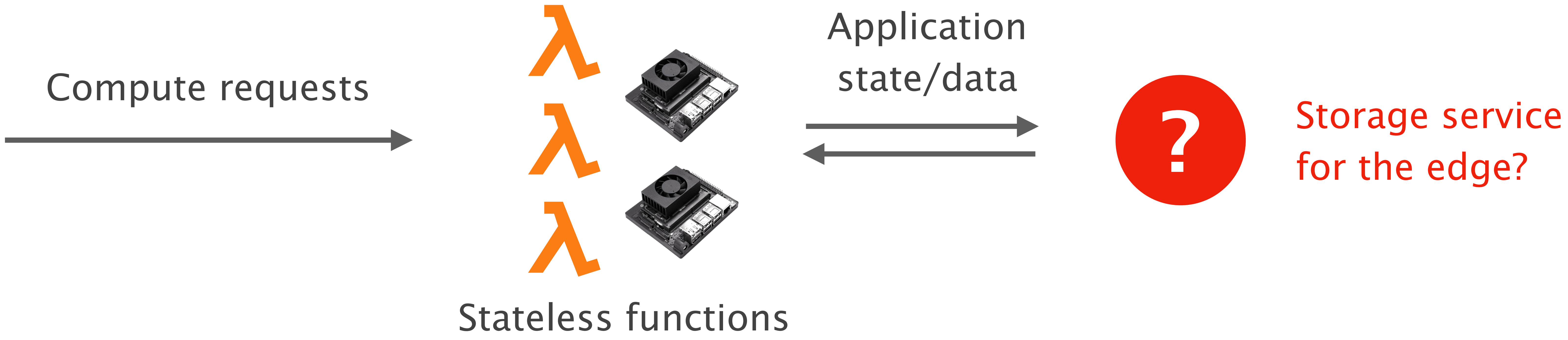
Principle — Decouple compute and storage for higher scalability and availability, and lower cost

Cloud solutions — HDFS, Amazon S3, Redis, Cassandra...

Serverless computing: function as a service



How to apply similar ideas to the edge?



Question: Can we just use cloud storage solutions at the edge?

Short answer is **NO**, because of the new challenges (distributed, heterogeneous, dynamic) imposed by the edge [Confais et al. CloudCom'16]

- High latency for strong consistency (multi-RTT)
- High cross-site traffic volume

What do we need to consider when designing an edge store ?

Abstractions/APIs
KV-pairs, graph-based, time-series

Locality
Replication, spatio-temporal encoding

Heterogeneity
Partial replica, TTL-based data eviction

Mobility
Session migration, replica placement

Failover
Zones, erasure encoding, CRDT

Scalability
Spatio-temporal hashing

Semantics
Context-/Location-based, consistency

Monitoring
Resource usage, dynamics, mobility

Where are we standing now?

Little to no support

	Abstraction/API	Locality	Heterogeneity	Mobility	Failover	Scalability	Semantics
PathStore	relational/CQL	✔	✘	◐	◐	✔	session/eventual
FogStore	key-value	✔	✘	✘	✔	✔	context-aware
DataFog	key-value	✔	✔	✘	◐	✔	eventual
RedWedding	CRDT	✔	✘	✘	⦸	✔	conflict-free
DPaxos	transactions	✔	✘	◐	✔	✔	quorum-based
EdgeCons	events	✔	✘	✘	✔	✔	quorum-based
TSDBs	time-series	✔	✘	✘	◐	✔	range, aggregate
Cachier	objects, contents	✔	✘	✘	⦸	⦸	N/A
Vision-specific	key-frames, features	✔	✘	✘	⦸	⦸	N/A

✔ Full support ◐ Partial support ✘ No support ⦸ Unknown

Various abstractions and semantics

Griffin: a multi-consistency hierarchical distributed storage service for edge computing

1 Multi-consistency declarative API

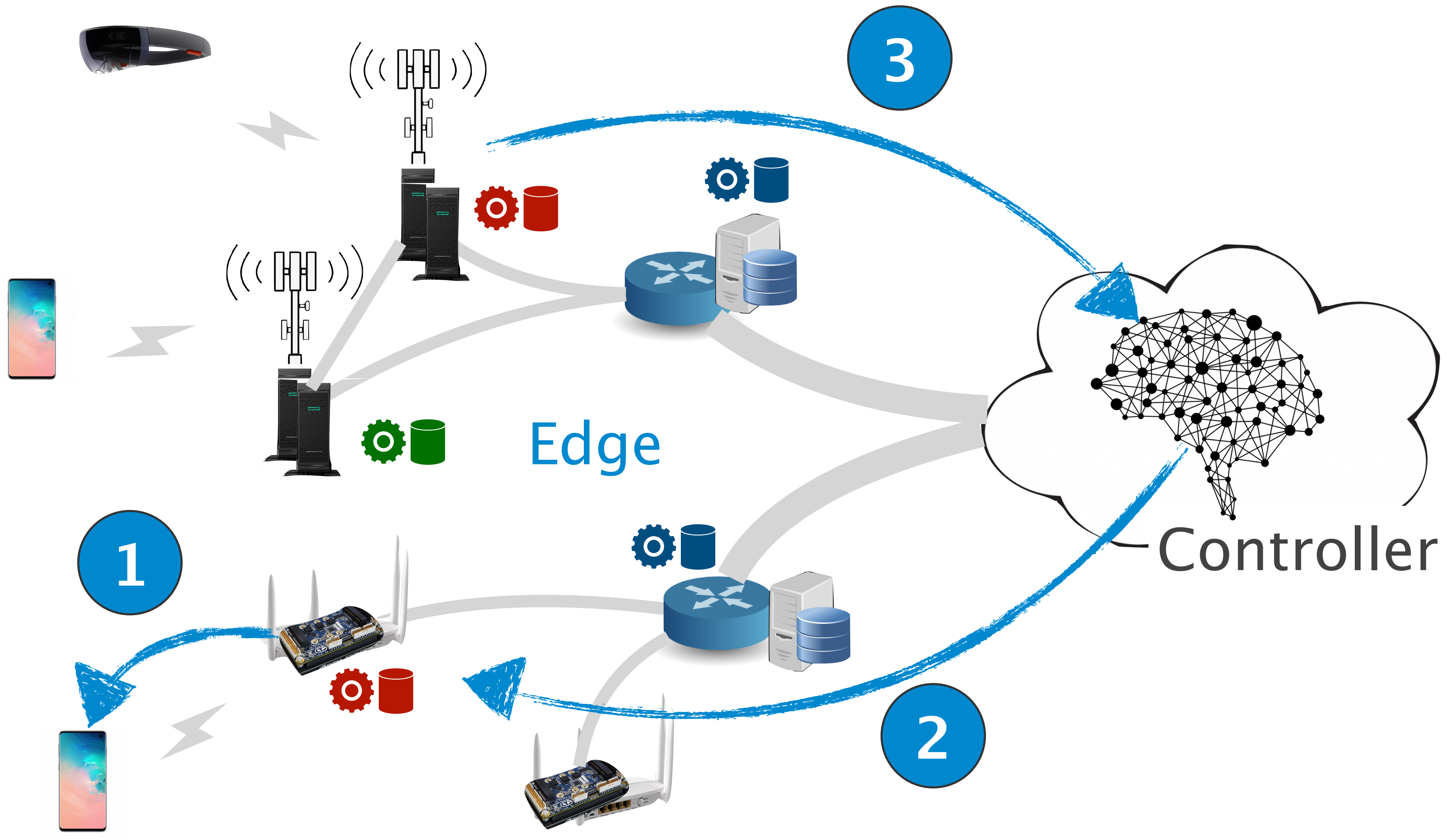
- Tradeoffs between latency and consistency [Terry et al. SOSP'13]
- Timestamp-based conflict resolution
- Reduce (de)serialization cost

2 Model-based resource management

- Graph-based models for heterogeneous resources
- Adaptive optimization mechanisms

3 Real-time monitoring

- Infrastructure-centric latency/resources monitoring
- Mobility monitoring/prediction



Key messages

- There is a need for a *first-class* service for data and state sharing for edge computing

- Edge data sharing requirements are *wide* and *diverse*

- Challenges to existing storage services when applied at the edge generate *opportunities* for a suitable edge store design

© Paul Dickenson

Discussion points

Usability

How should we make edge application *develops*' life easier? What abstractions to facilitate state externalization?

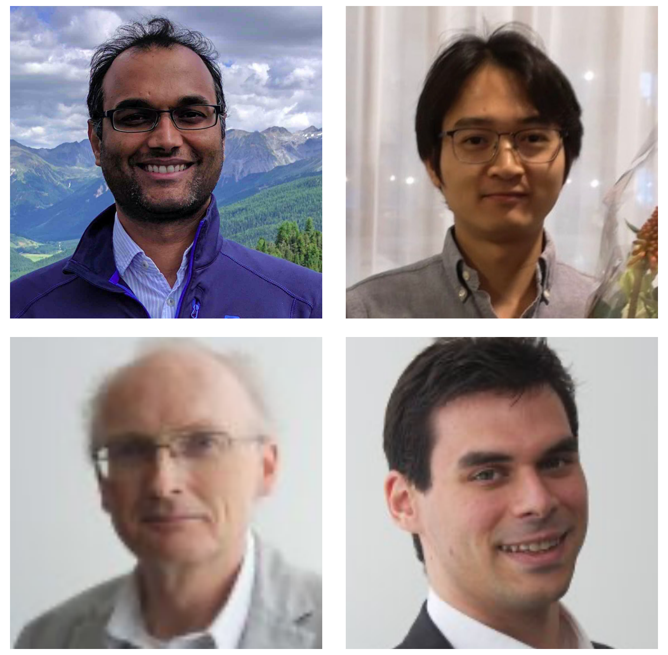
Management

Is it realistic to monitor and model the edge environment with all its *complexities*? How resources should be *shared* among different services including edge store?

Incentives

How to design a *cooperation* framework for multiple edge providers (e.g., like peering on Internet)? How to handle *privacy*-related concerns?

Sharing and Caring of Data at the Edge



Animesh Trivedi*, **Lin Wang***,
Henri Bal, Alexandru Iosup

USENIX HotEdge'20
June 25, 2020

VU VRIJE UNIVERSITEIT AMSTERDAM

TECHNISCHE UNIVERSITÄT DARMSTADT