Edge Native

Bringing the Cloud-Native Development and Operations Experience to the Edge

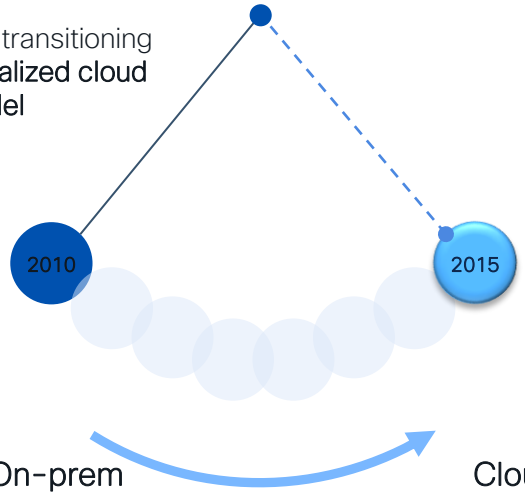
Frank Brockners, Distinguished Engineer, Cisco

Cisco
Emerging Technologies
and Incubation

KubeCon | CloudNativeCon
North America 2022

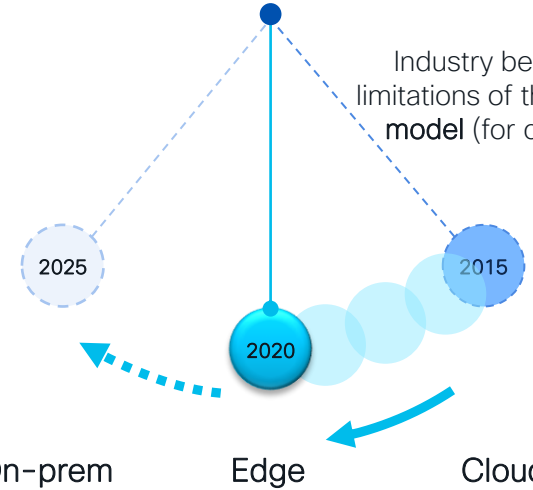
Cloud disrupted the on-prem paradigm…

Industry starts transitioning towards **centralized cloud model**



…Edge emerging as the next paradigm

Industry beginning to realize limitations of the **centralized cloud model** (for certain use cases)



Reduced Data Transit Cost

Data-Gravity: Local processing with dedicated edge compute limits need for expensive cloud storage and backhaul

Video analytics and streaming, industrial IoT

Real-time Performance

Low latency and high throughput for superior application experience and real time insights

AR/VR, Cloud Gaming Video/Content Streaming, Autonomous vehicles, Smart grid, Remote monitoring

Privacy and Regulations

Caters to data sovereignty & compliance needs with distributed model

GDPR, Security Regulation

Can we make the “Edge” feel like the cloud from a development and operations perspective?

Easy

Hide the “Edge’s” complexities/specifics

Fast

Deploy in seconds rather than months

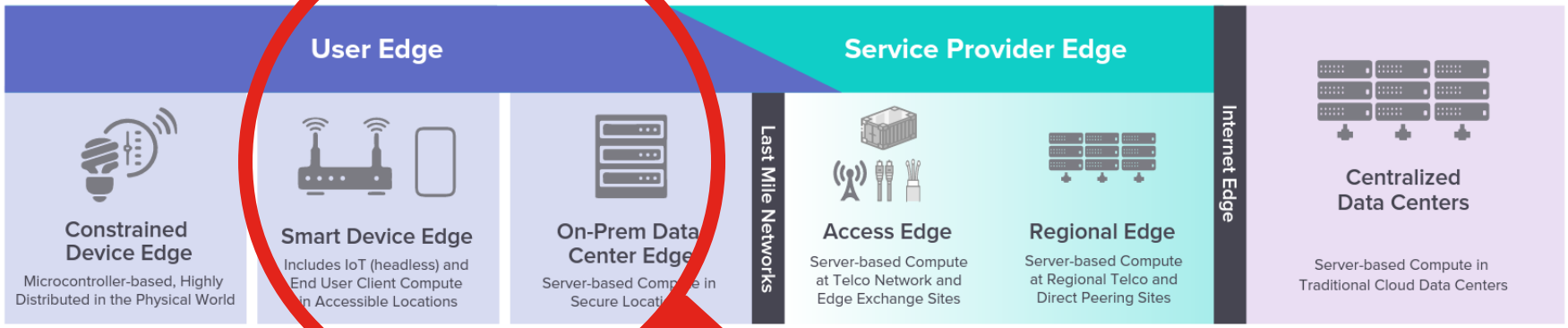
Familiar

Leverage and integrate with common tool chains

Which “Edge”?

Homogenous
DevOps cloud operating model
Resource rich

Cloud-Out



Edge-In

Location-aware, Heterogeneous
Non-IT users & operators
Resource constrained, constrained connectivity

Source: LF-Edge Taxonomy

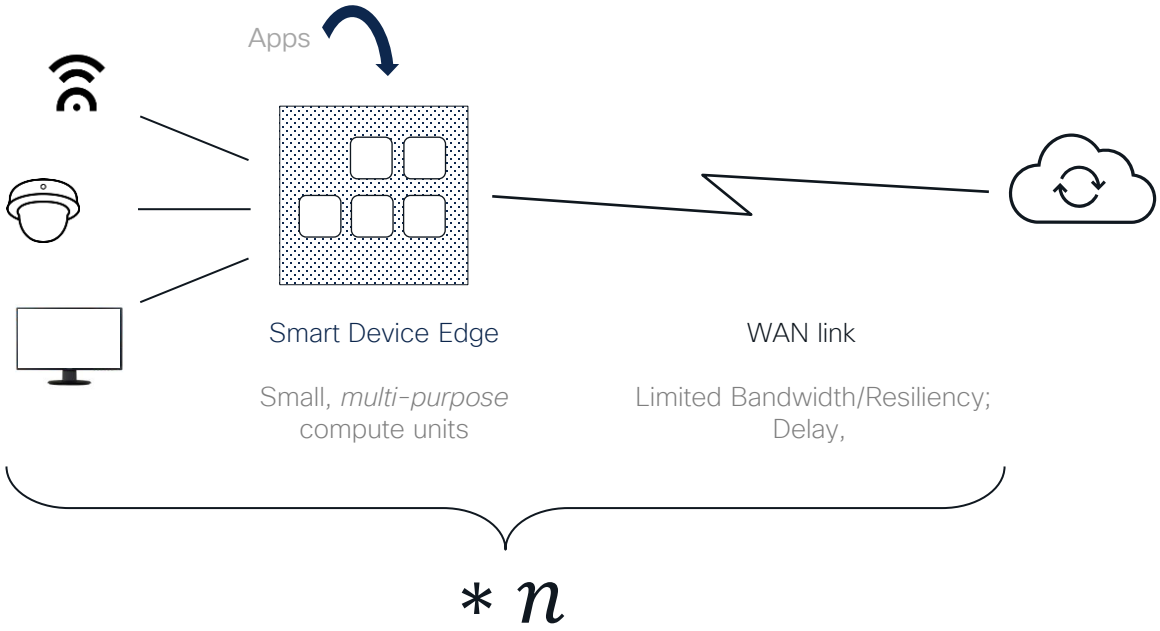
Smart Device Edge Transitions

Single purpose-fit compute unit  Multi-purpose cluster of small computers

"Vertical" point solutions  "Horizontal" platform

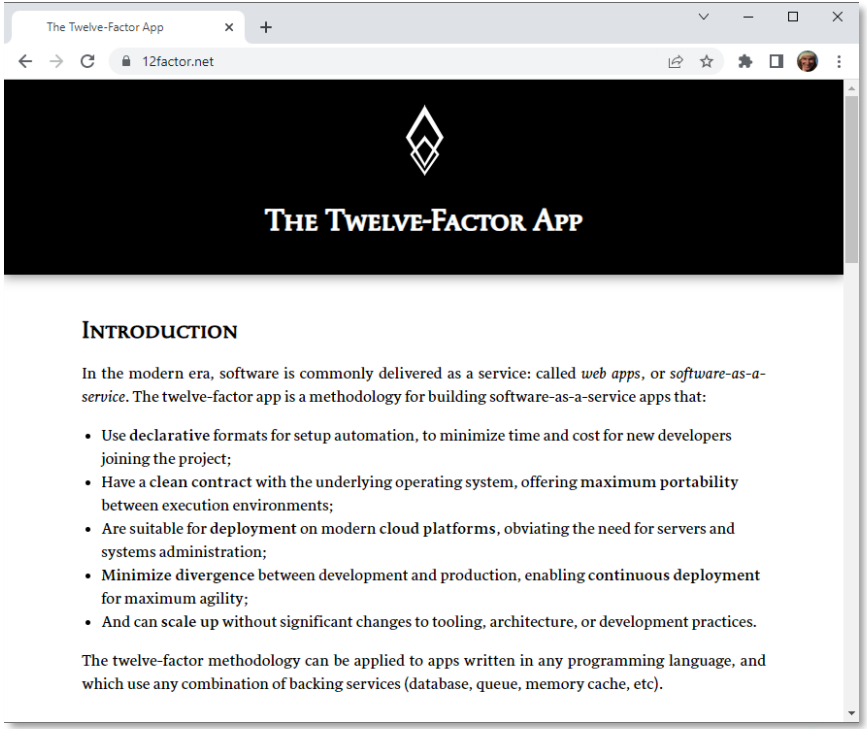
Manual deployment  Automated, aaS

Smart Device Edge – Deployment Scenario



From Cloud-Native to Edge-Native…

… evolving the
“Twelve Factors”.



The Twelve Factors

I. Codebase
One codebase tracked in revision control, many deploys

II. Dependencies
Explicitly declare and isolate dependencies

III. Config
Store config in the environment

IV. Backing services
Treat backing services as attached resources

V. Build, release, run
Strictly separate build and run stages

VI. Processes
Execute the app as one or more stateless processes

VII. Port binding
Export services via port binding

VIII. Concurrency
Scale out via the process model

IX. Disposability
Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity
Keep development, staging, and production as similar as possible

XI. Logs
Treat logs as event streams

XII. Admin processes
Run admin/management tasks as one-off processes

Source: 12factor.net

The Twelve Factors
at the Edge

I. Codebase
One codebase tracked in revision control, many deploys

II.e. Dependencies and Policies
Explicitly declare and isolate dependencies and policies – edge-local and cloud

III. Config
Store config in the environment

IV.e. Backing services
Treat any backing service as an attached, potentially remote, resource

V. Build, release, run
Strictly separate build and run stages

VI. Processes
Execute the app as one or more stateless processes

VII. Port binding
Export services via port binding

VIII.e. Concurrency / Scale
Scale by the site model and the process model

IX.e. Disposability
Robustness by site: Fast startup, graceful shutdown, declarative desired state

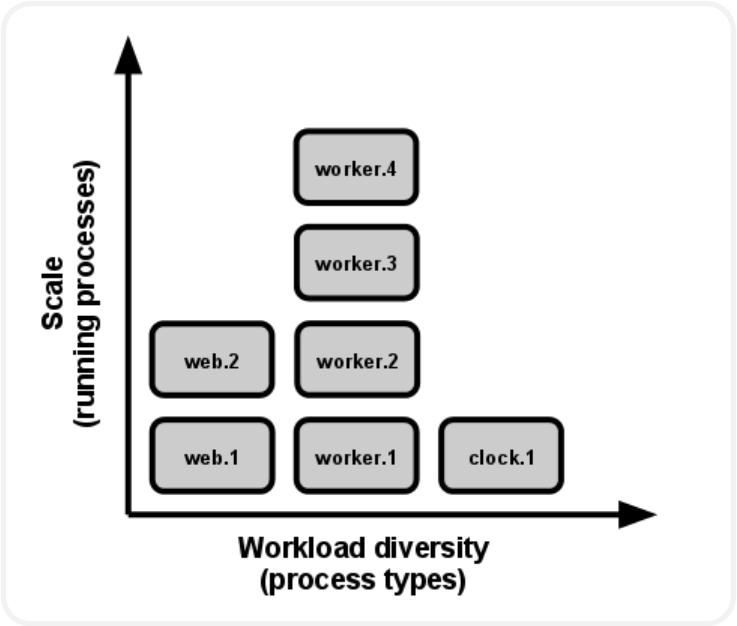
X. Dev/prod parity
Keep development, staging, and production as similar as possible

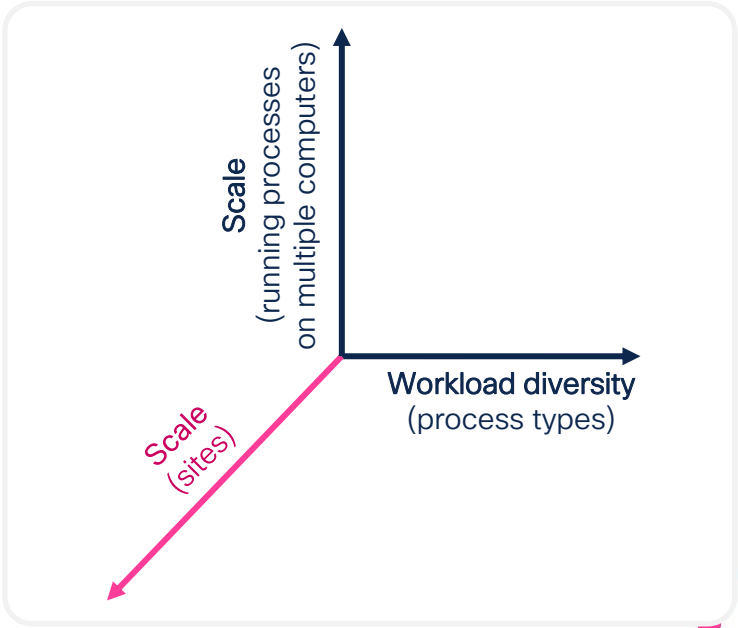
XI.e. Logs
Treat logs and metrics as on-demand streams, use metrics whenever feasible

XII. Admin processes
Run admin/management tasks as one-off processes

VIII.e. Concurrency / Scale

Sites (locations) as the new key scale dimension for Edge deployments

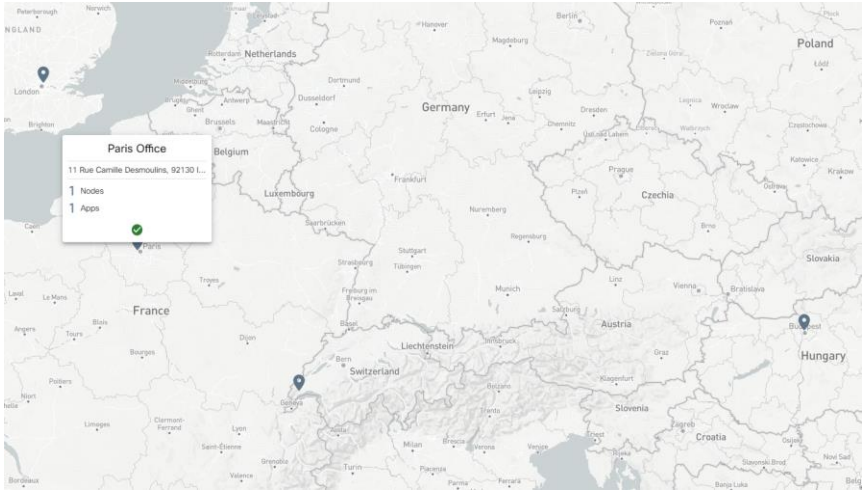




Sites

Sites are an abstract notion of “location”

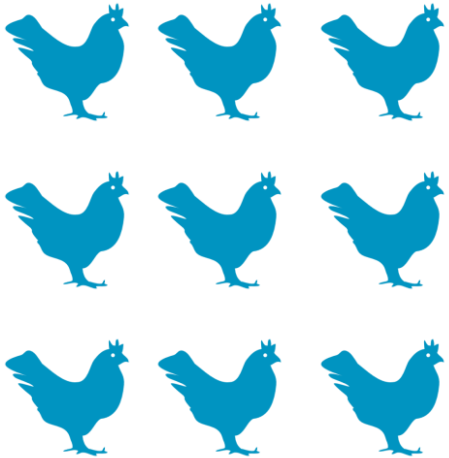
- Sites represent a physical or logical grouping (i.e., could also be “dev team”) of compute nodes that run one or multiple processes/ applications.
- Applications are logically deployed to Sites.
- Sites are composed of small *multi-purpose* compute nodes.



IX.e. Disposability

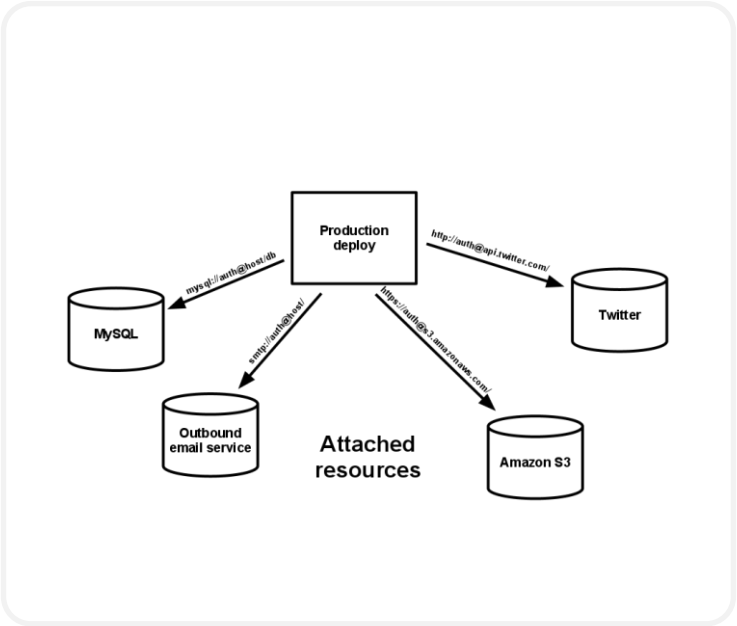
Robustness by site – Sites and nodes are cattle

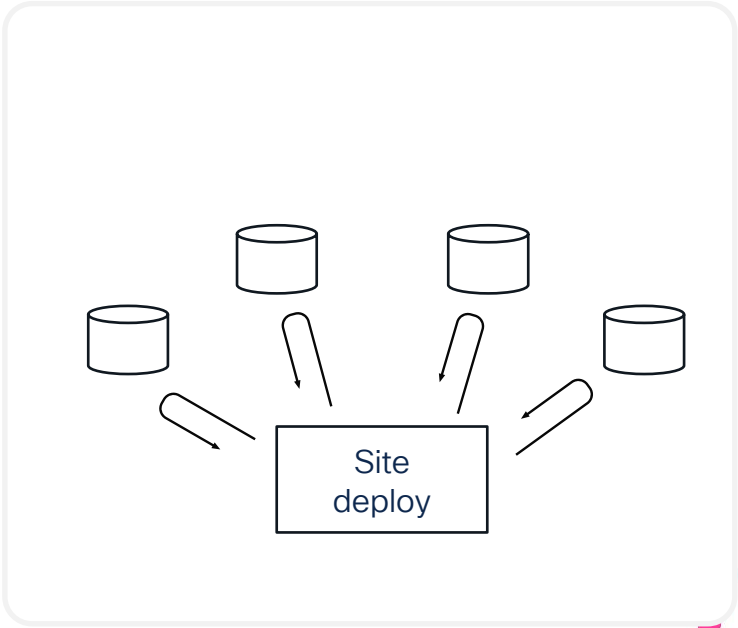
- Sites and nodes are disposable.
- Sites and nodes, like processes, can be started or stopped at a moment's notice.
- Neither sites, nodes, nor the processes that they run, are not "debugged" in case they do not function properly:
  "Factory reset – re-bootstrap" to resolve issues.



IV.e. Backing services

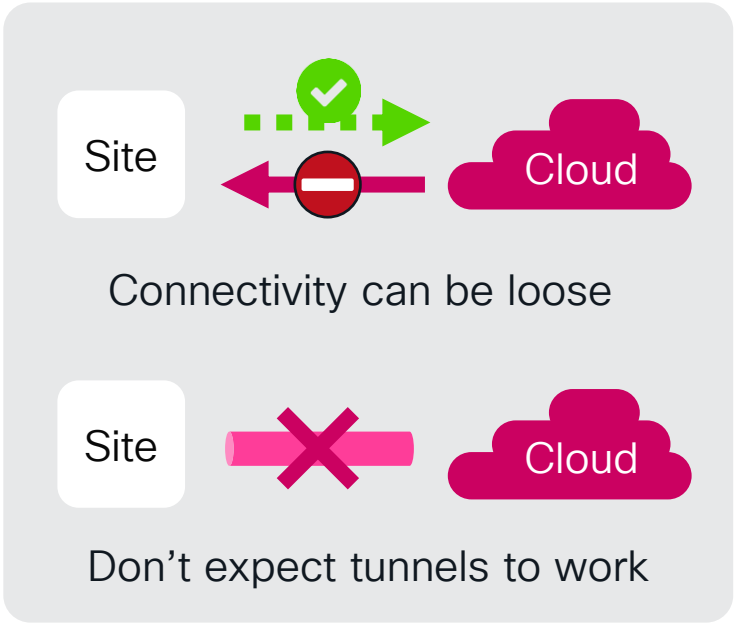
Treat any backing services as an attached, potentially remote, resource.

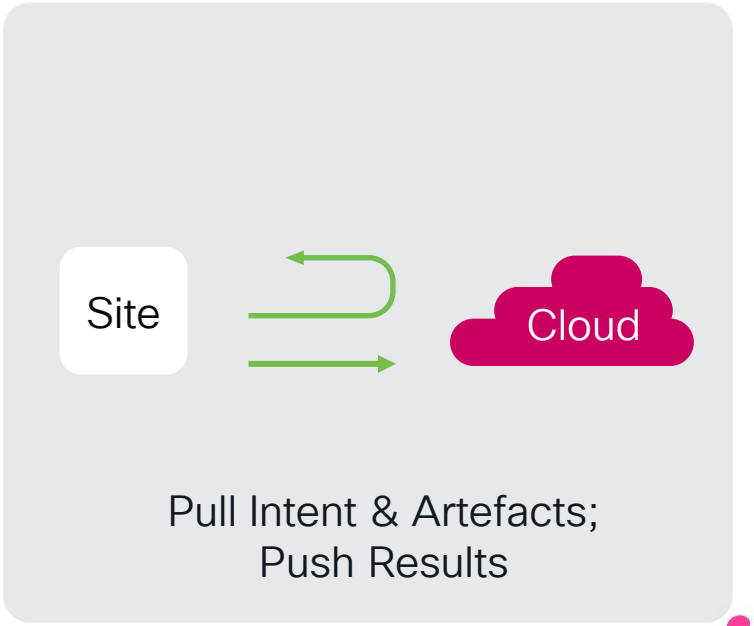




IV.e. Backing services

Pull, don’t push. Don’t expect tunnels. 100% declarative

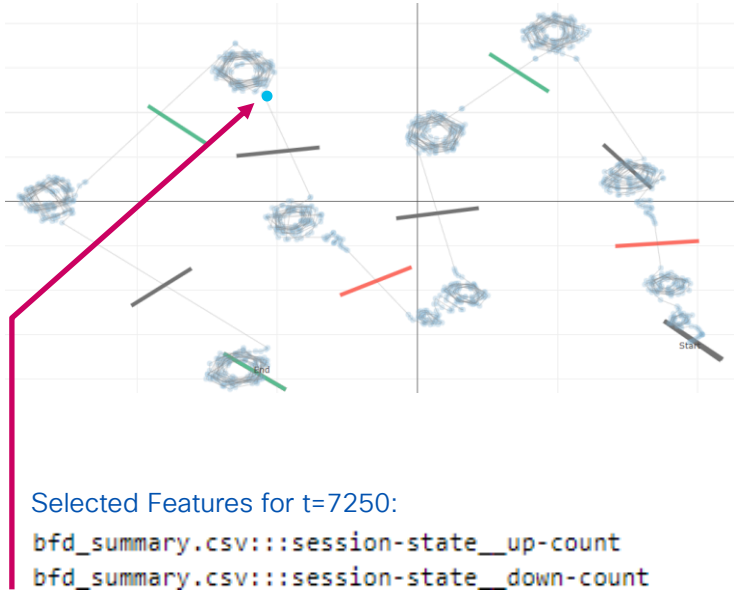




XI.e. Logs

Stream metrics and logs

- Stream/push metrics and logs
- On-demand retrieval of logs
- Metrics help measure component functionality, and combined with additional logic can raise alerts
- Focus on “actionable” metrics: Send “information”, not “data”



XI.e. Logs

DESTIN: Detecting State Transitions in Network elements

Parisa Foroughi, Wenqin Shao, Frank Brockners, Jean-Louis Rougier
Proceedings of the 17th IFIP/IEEE International Symposium on Integrated Network Management, 2021 (paper)

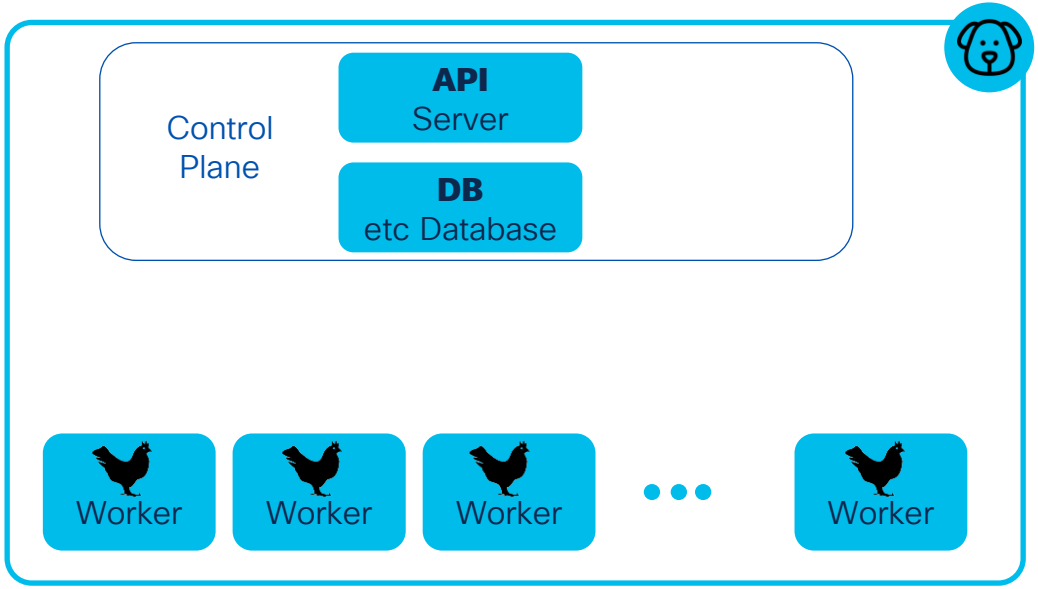
Semantic feature selection for network telemetry event description

Thomas Feltin, Parisa Foroughi, Wenqin Shao, Frank Brockners, Thomas H. Clausen
IEEE/IFIP Network Operations and Management Symposium, AnNet Workshop, 2020 (paper)

Edge Native Applied:
Deployment & Operations

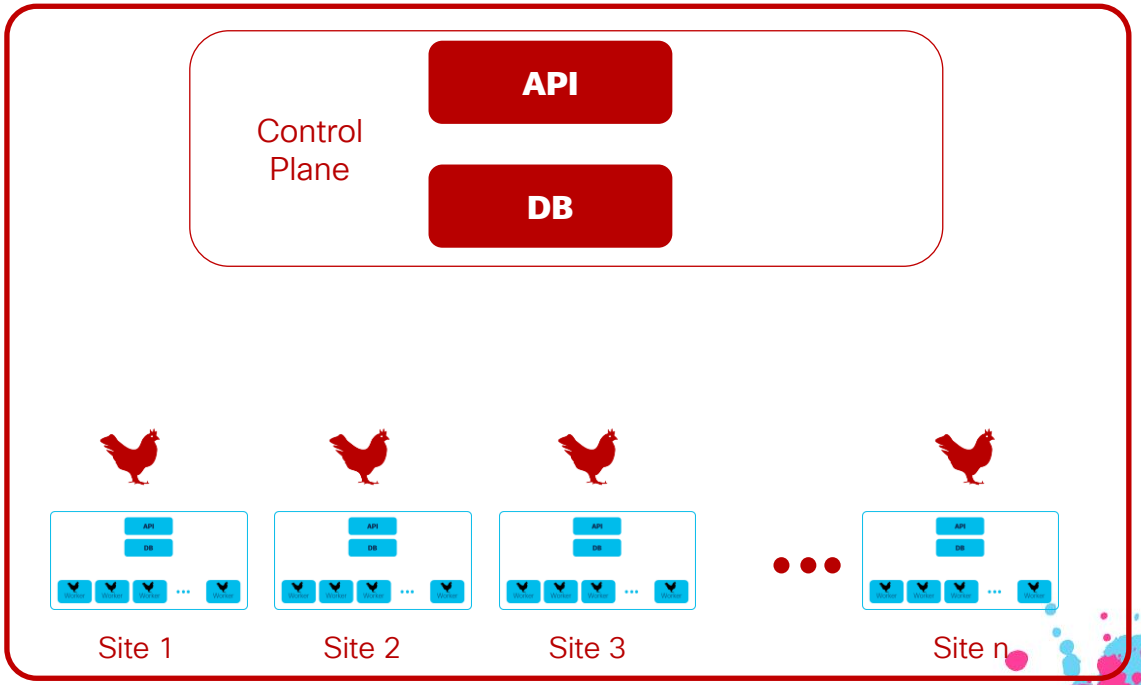
Kubernetes over-simplified:

Clusters are **Pets**, **Workers** are **Cattle**



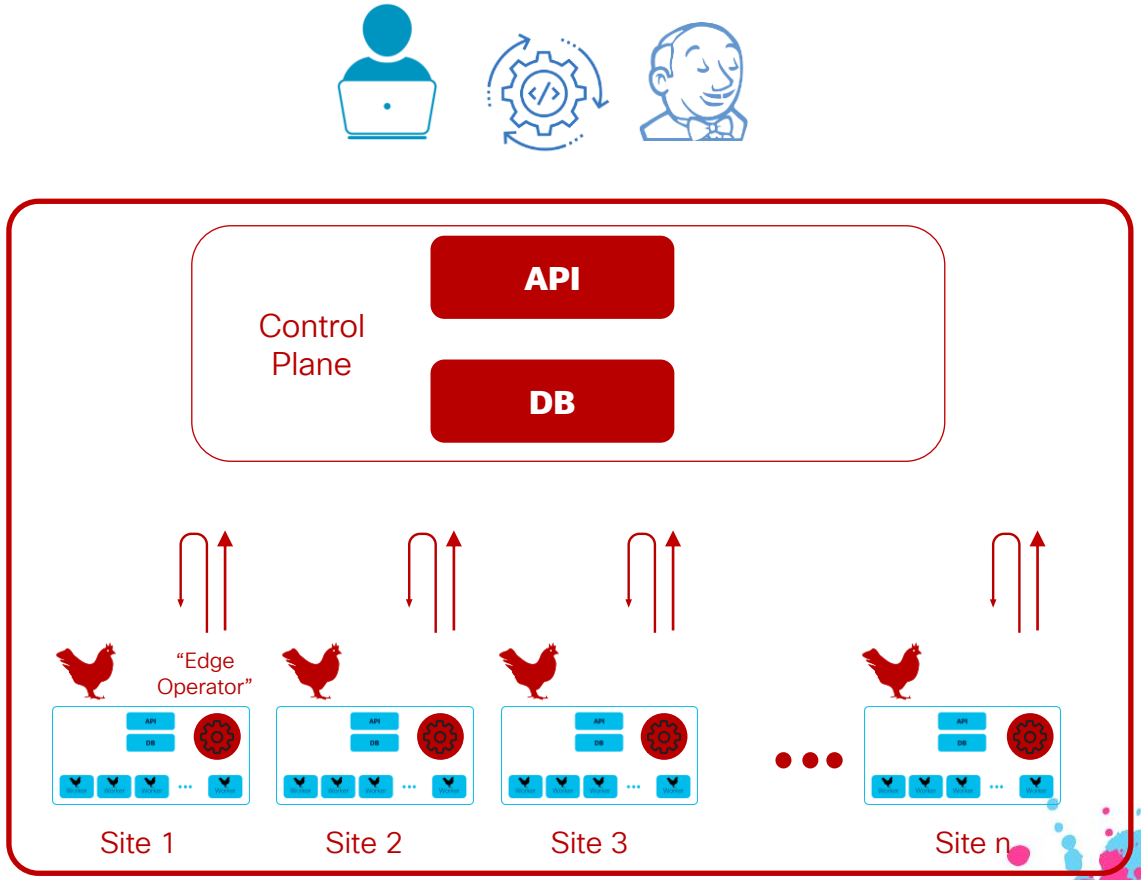
Edge Native:

Clusters are now “Cattle” and can fail any time.



Sites Pull CICD/Admin Defined State;

Sites Push Results/State



Some eye candy: Deployment example

I'd like to **roll out** a new **site** and **deploy** an **app**

Apps – Applications deployed to Sites

Sites – Groups of Nodes

AppNodes – Small Computers

Example AppNode

RPi4B - Booted with OS and Great Bear Agent

S/N Q3CA-64X5-4QBK

Great Bear

Great Bear | App Control UI

ref-dashboard.prod.gbear.scratch.eticloud.io/gbear/welcome

cisco Great Bear | IT

Welcome!

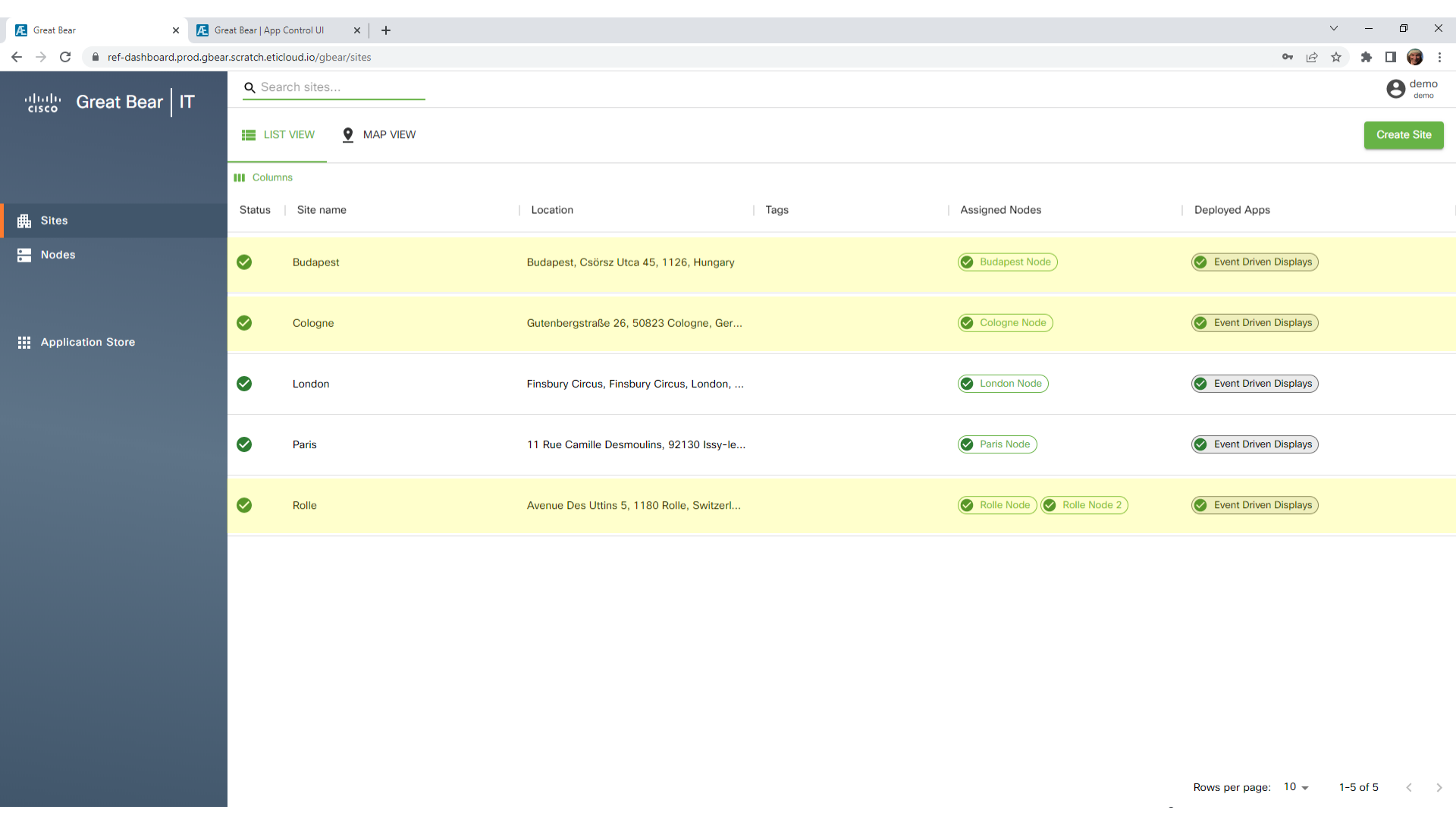
Great Bear - Apps as a Service

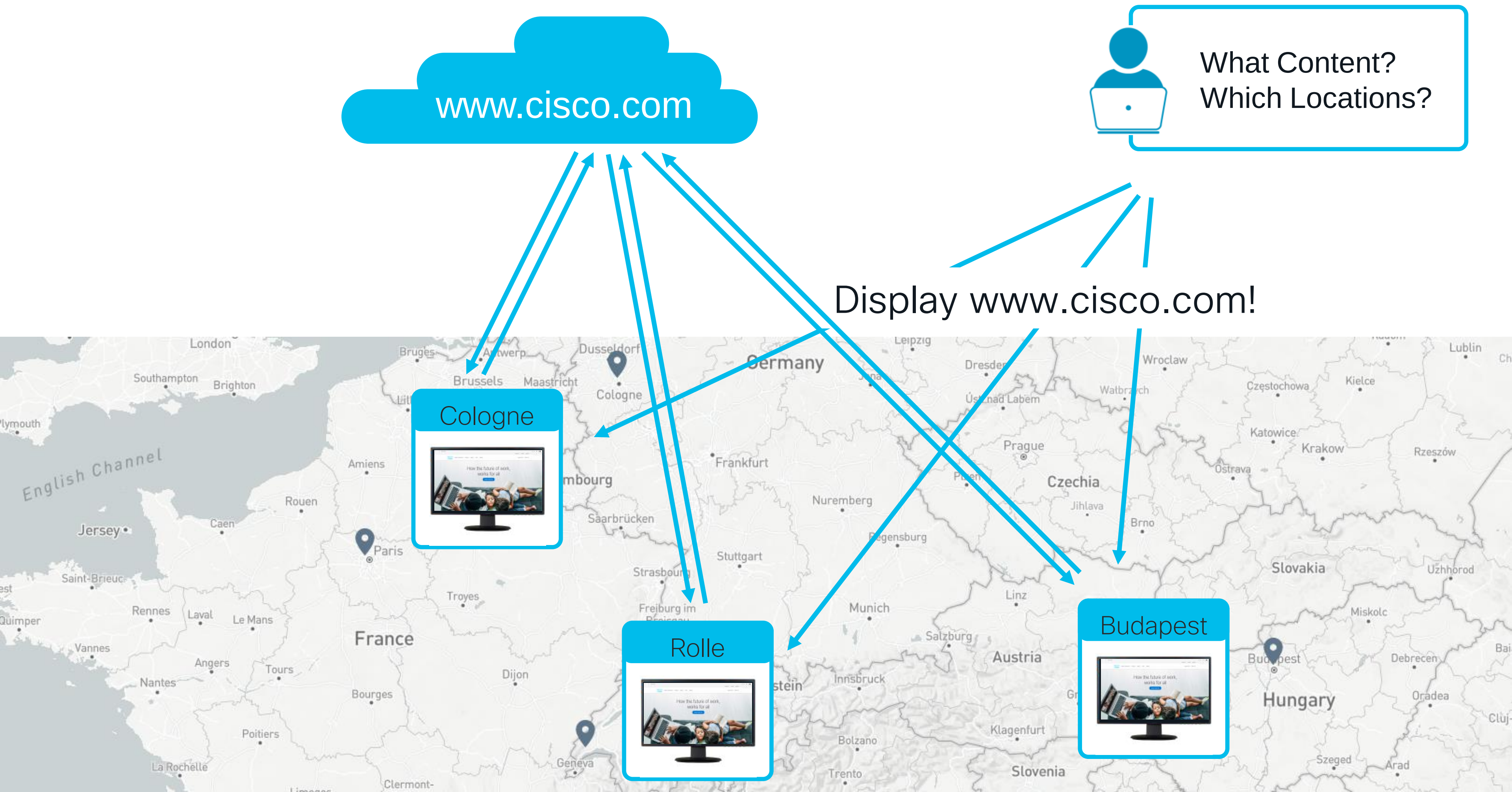
Great Bear offers Apps as a Service for users and an associated SDK for developers, greatly simplifying App
lifecy nt and development at the
ge.

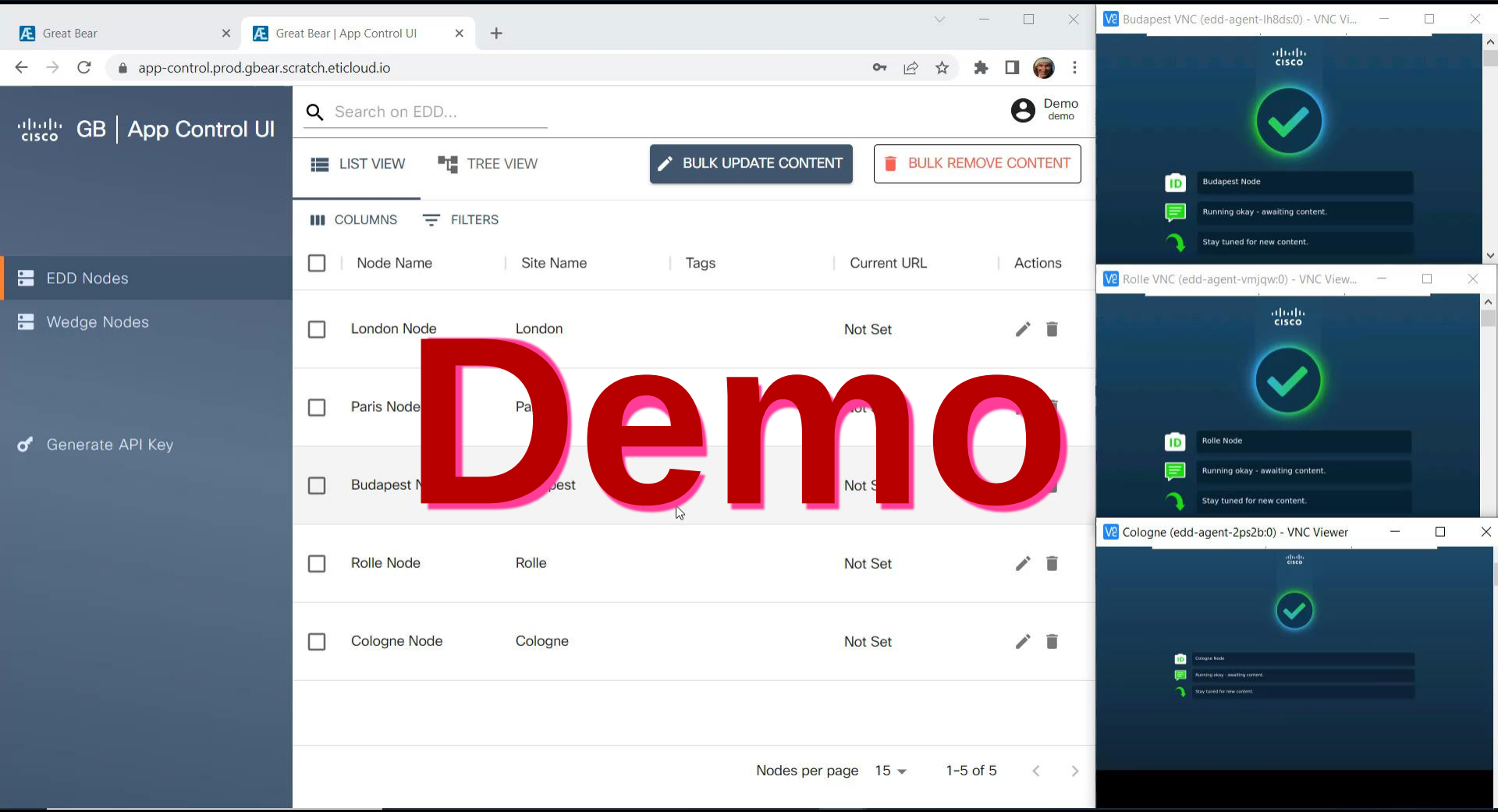
Demo

“Event Driven Display”

Render and display content locally at the Edge for Digital Signage, Video Surveillance, …


Great Bear
Great Bear | App Control UI
ref-dashboard.prod.gbear.scratch.eticloud.io/gbear/sites
Great Bear | IT
Search sites...
demo
demo
LIST VIEW
MAP VIEW
Create Site
Columns
Sites
Nodes
Application Store
Status
Site name
Location
Tags
Assigned Nodes
Deployed Apps
Budapest
Budapest, Csörsz Utca 45, 1126, Hungary
Budapest Node
Event Driven Displays
Cologne
Gutenbergstraße 26, 50823 Cologne, Ger...
Cologne Node
Event Driven Displays
London
Finsbury Circus, Finsbury Circus, London, ...
London Node
Event Driven Displays
Paris
11 Rue Camille Desmoulins, 92130 Issy-le...
Paris Node
Event Driven Displays
Rolle
Avenue Des Uttins 5, 1180 Rolle, Switzerl...
Rolle Node
Rolle Node 2
Event Driven Displays
Rows per page: 10
1-5 of 5


www.cisco.com
What Content?
Which Locations?
Display www.cisco.com!
Cologne
Rolle
Budapest


Great Bear
Great Bear | App Control UI
app-control.prod.gbear.scratch.eticloud.io
GB | App Control UI
Search on EDD...
Demo
demo
LIST VIEW
TREE VIEW
BULK UPDATE CONTENT
BULK REMOVE CONTENT
COLUMNS
FILTERS
Node Name
Site Name
Tags
Current URL
Actions
EDD Nodes
Wedge Nodes
Generate API Key
London Node
London
Not Set
Paris Node
Budapest Node
Not Set
Rolle Node
Rolle
Not Set
Cologne Node
Cologne
Not Set
Nodes per page
15
1–5 of 5
Demo
Budapest VNC (edd-agent-lh8ds:0) - VNC Vi...
Budapest Node
Running okay - awaiting content.
Stay tuned for new content.
Rolle VNC (edd-agent-vmjqw:0) - VNC View...
Rolle Node
Running okay - awaiting content.
Stay tuned for new content.
Cologne (edd-agent-2ps2b:0) - VNC Viewer

Edge Native Applied:
Services & Development

Easy

Hide the “Edge’s” complexities/specifics

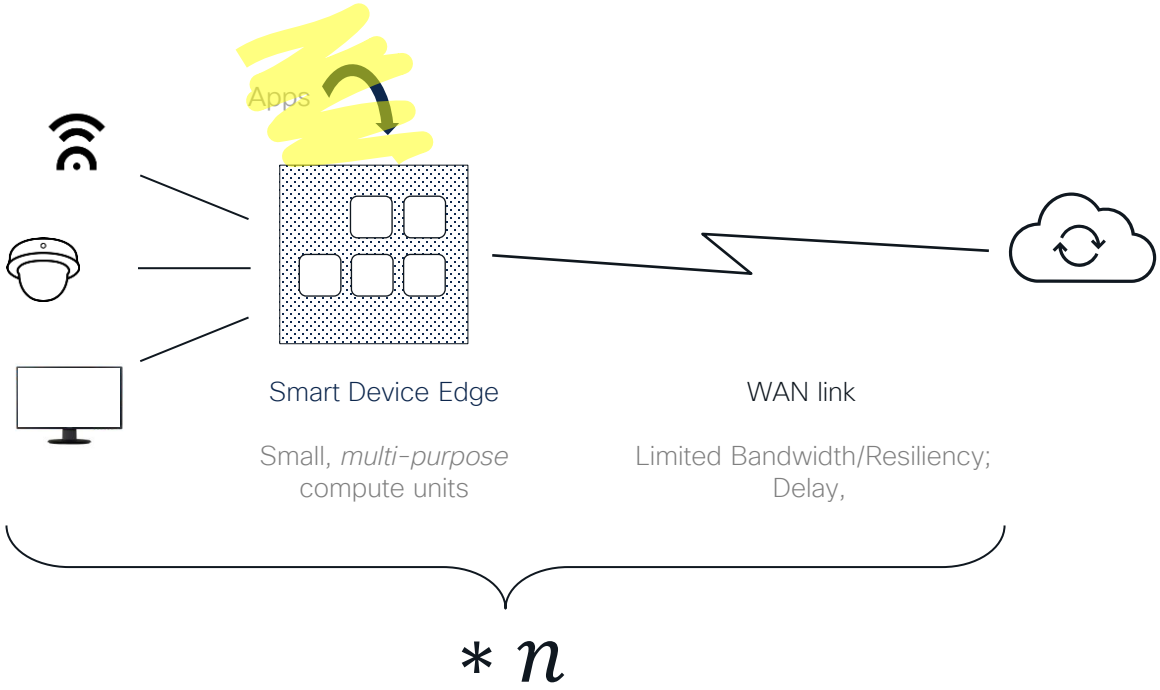
Fast

Deploy in seconds rather than months

Familiar

Leverage and integrate with common tool chains

Smart Device Edge – Deployment Scenario



Common Needs Across Use-Cases

Edge-AI

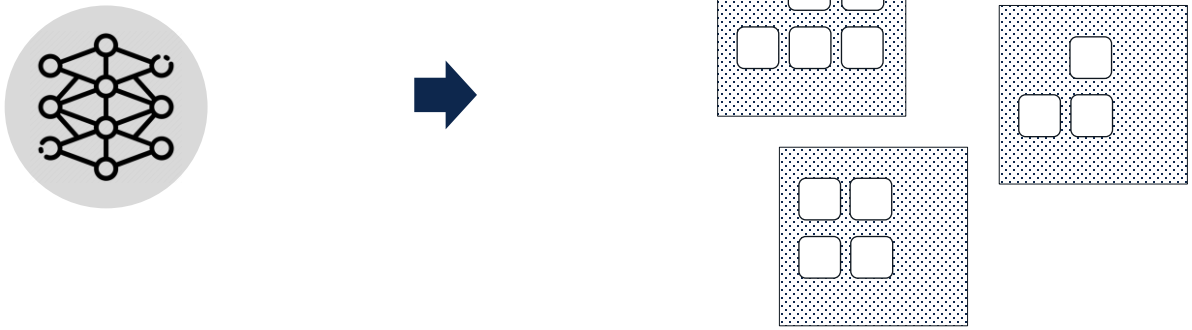
Fit AI models to the resources available at the edge for inference; Federate learning

Data Management

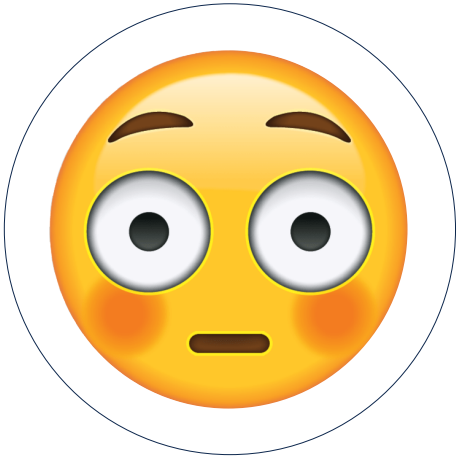
Pipelined processing, on-demand invocation of workloads; Data caching and storage

Edge Rendering/IO

Render and display streaming and static data locally


AI Model
Deploy
Sites

I have developed 2 new AI models.



I have developed 2 new AI models.

The first one requires 100 MB,
but is too large to fit onto one of my compute nodes…

I have developed 2 new AI models.

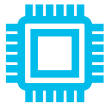
The first one requires 100 MB,
but is too large to fit onto one of my compute nodes…

The second one runs with 2 FPS on my node, but I need 5 FPS…

2 FPS **5** people detected

5 FPS **6** people detected

Inference at the Edge:
Dealing with constrained compute resources



Hardware Acceleration
(specific/dedicated compute)

Adding specialized hardware to the edge network to support the DNN computation (GPU, VPU, TPU, etc.)

Model Compression
(simplify the structure)

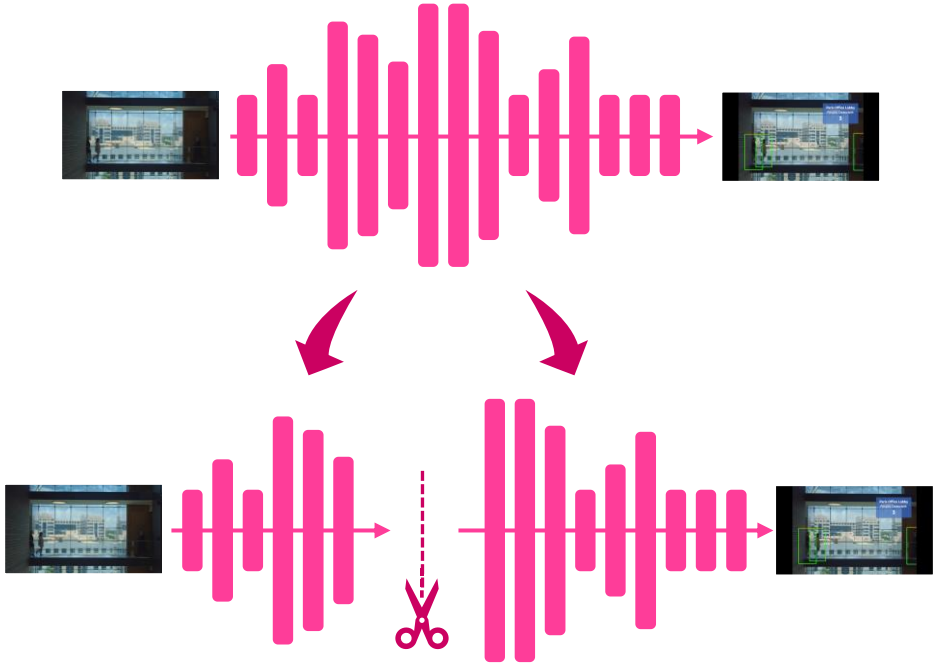
Quantization, layer reduction, operator fusion, graph optimization,…: Optimize the model size/structure for lighter inference to run faster on edge devices while optimizing resource utilization

Model Distribution
(split, load-balance)

Partitioning a model and sharing the computation over multiple nodes; Load balancing input data (e.g., frames) between multiple nodes

Distribute a model over multiple nodes to improve performance

Computing the optimal split of the Deep Neural Network and distributing the workloads

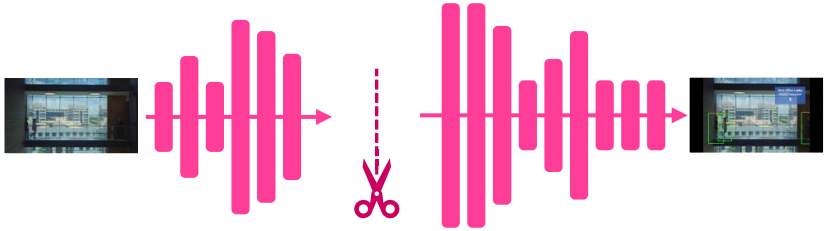


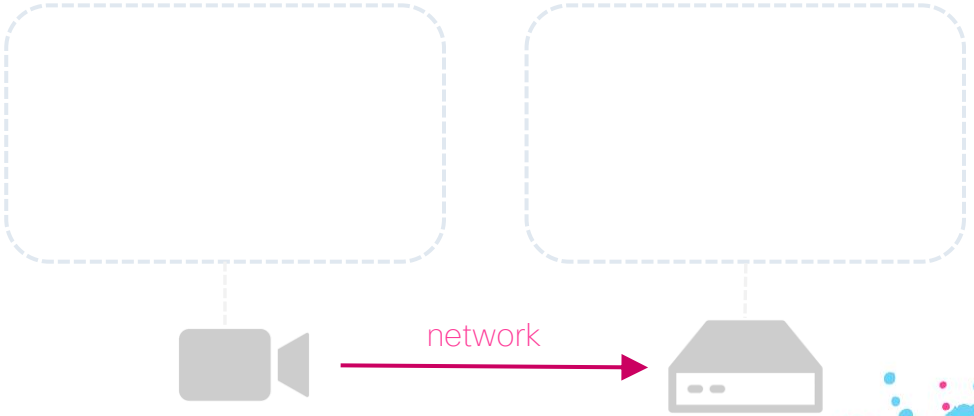
Distributed AI Algorithms consider

Available resources at the edge nodes

Network bandwidth constraints

Data transfer between split neural network layers





ONNX

~ **1.1** FPS

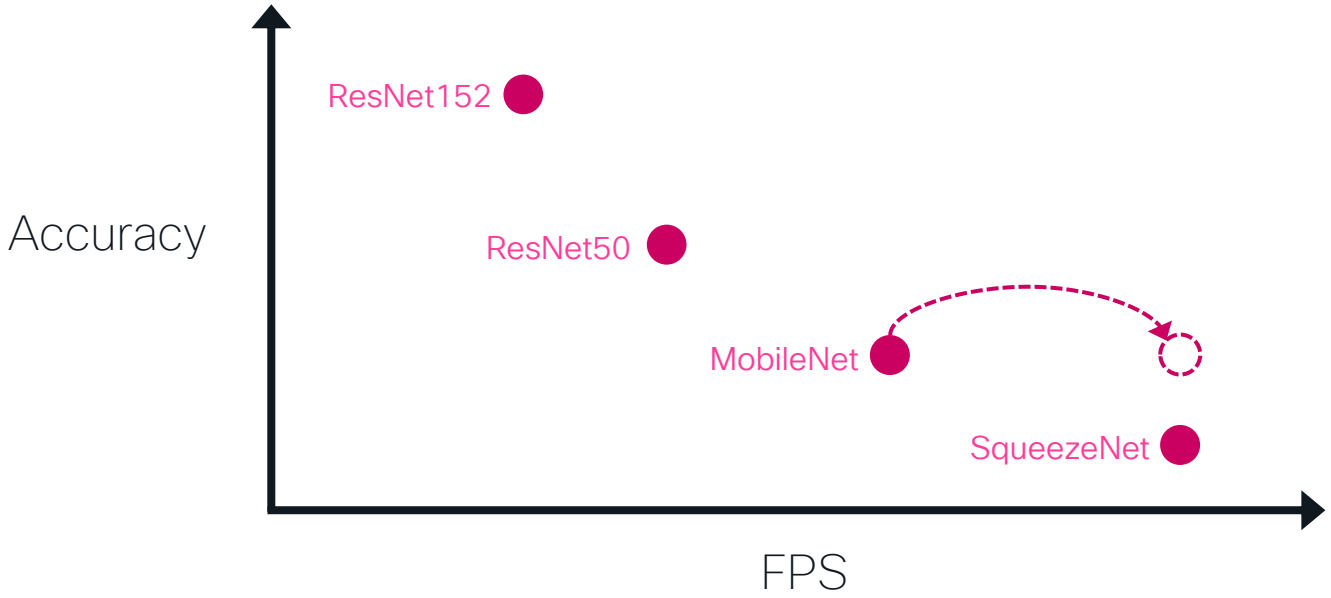
ONNX ONNX

~ **1.7** FPS

ONNX TFLite

~ **2.4** FPS

By splitting and distributing the model, we can help improve FPS without sacrificing accuracy

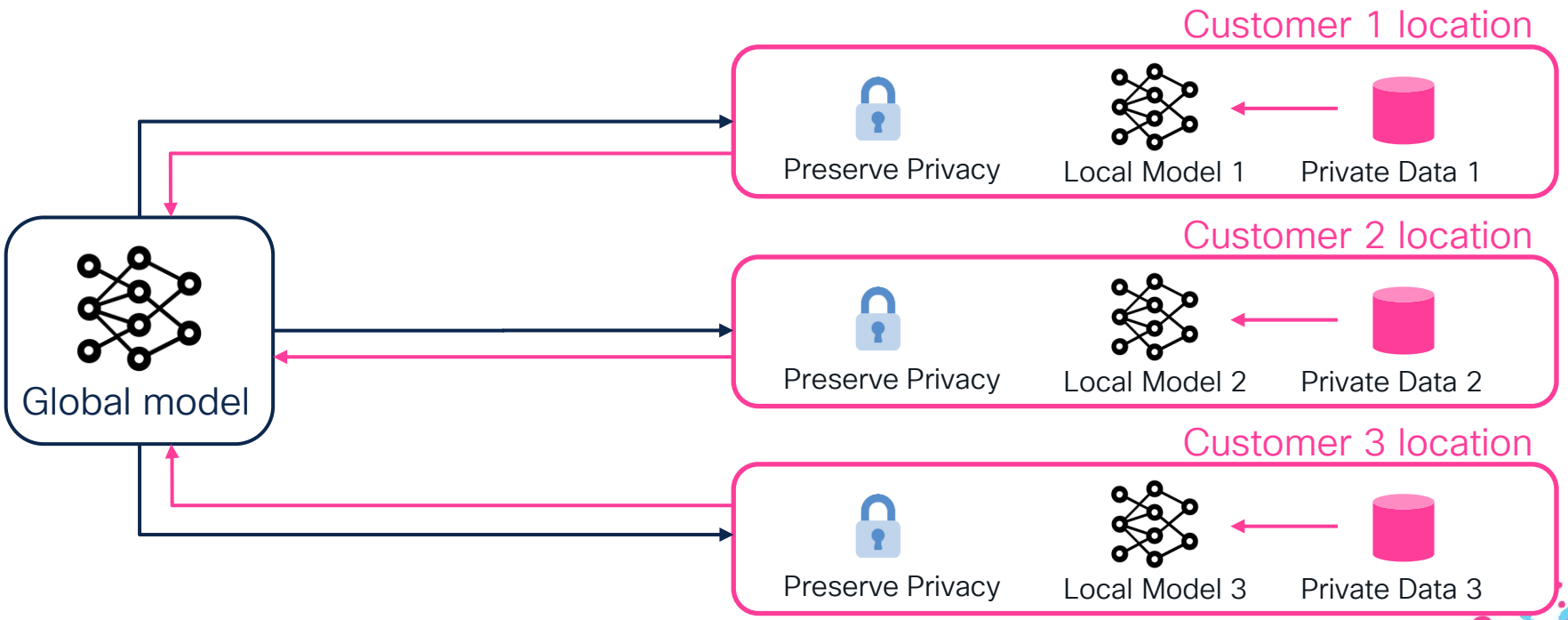


What about model training?

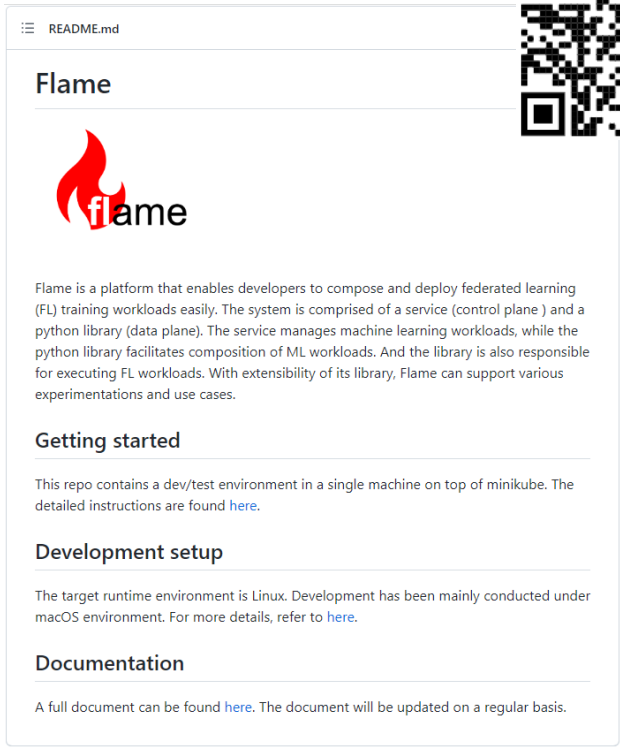
I like to train an AI model, but can’t get access to my customers data

Federated Learning!

Federated Learning



Compose and deploy federated learning training workloads easily across many edges


README.md

Flame

Flame is a platform that enables developers to compose and deploy federated learning (FL) training workloads easily. The system is comprised of a service (control plane) and a python library (data plane). The service manages machine learning workloads, while the python library facilitates composition of ML workloads. And the library is also responsible for executing FL workloads. With extensibility of its library, Flame can support various experimentations and use cases.

Getting started

This repo contains a dev/test environment in a single machine on top of minikube. The detailed instructions are found here.

Development setup

The target runtime environment is Linux. Development has been mainly conducted under macOS environment. For more details, refer to here.

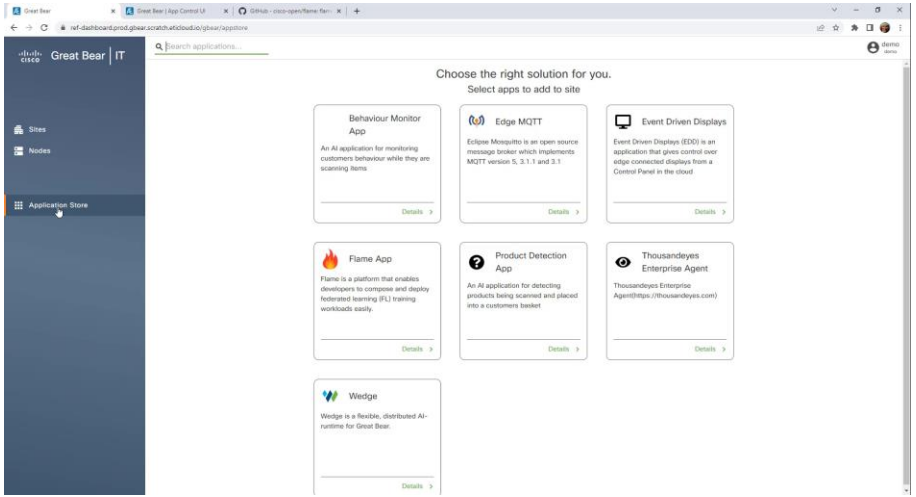
Documentation

A full document can be found here. The document will be updated on a regular basis.

github.com/cisco-open/flame

Federated Learning fits the Edge-Native principles very well

- Pull the global model
- Push model updates
- Keep data private per location

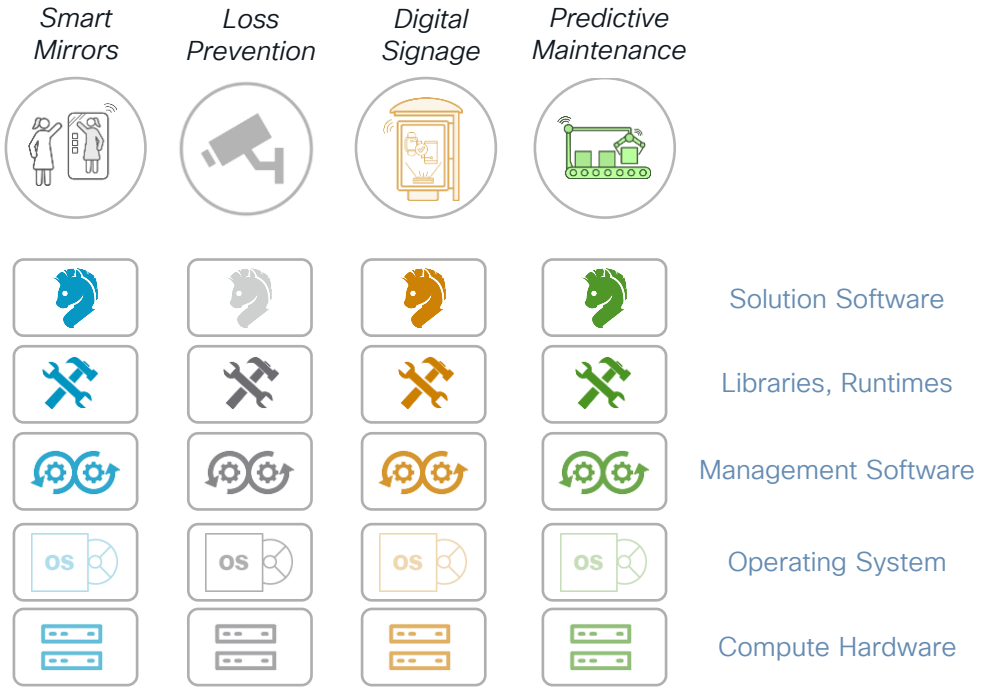


In Summary

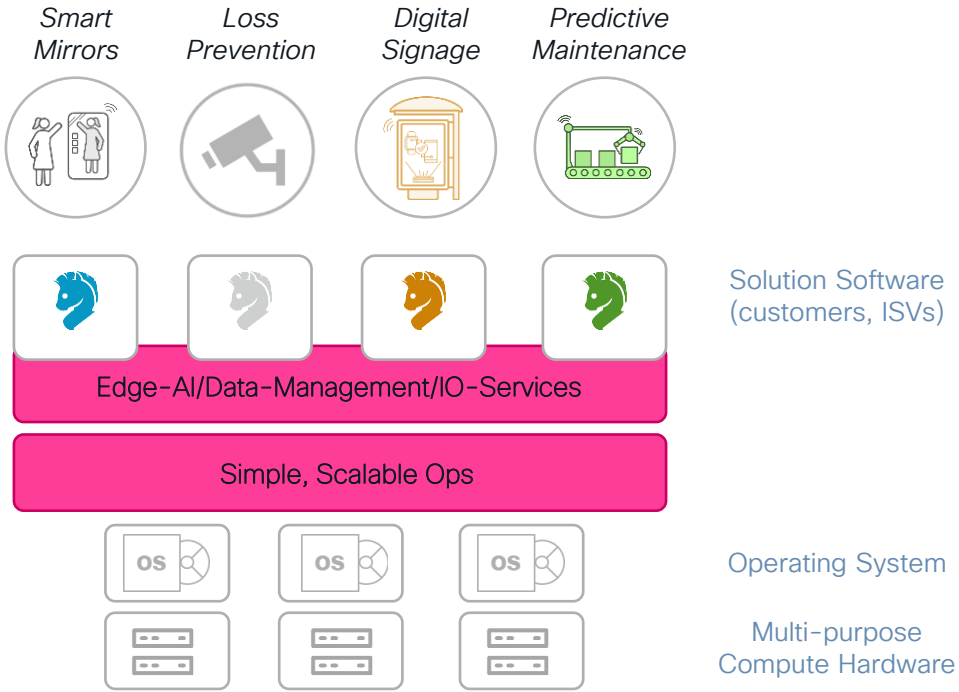
Edge-Native enables a “cloud-like” experience at the Edge…

… and enables a transition

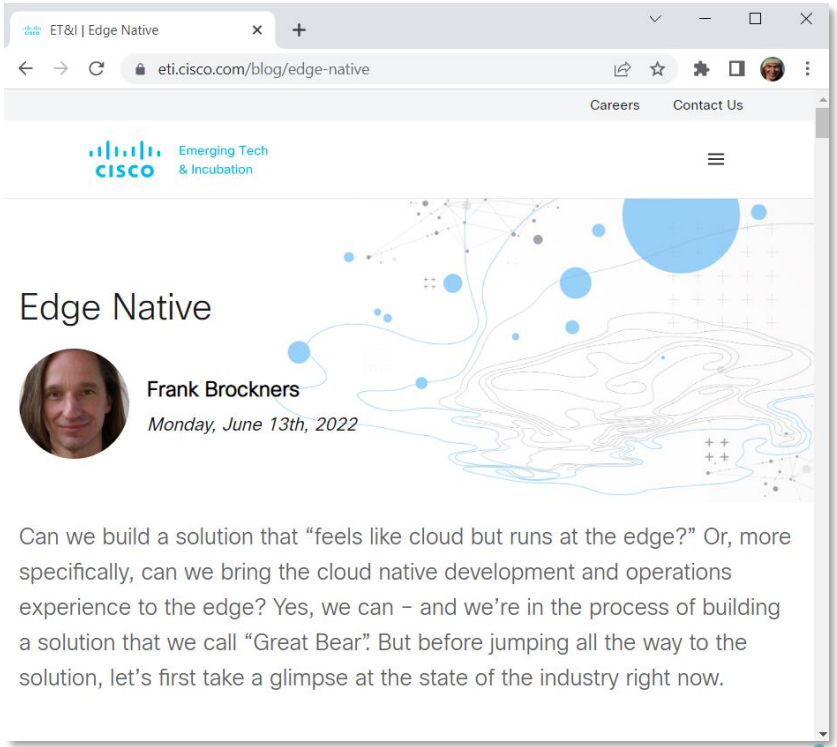
Point Solutions



Platform



eti.cisco.com/blog/edge-native



Thank You

Cisco
Emerging Technologies
and Incubation

KubeCon | CloudNativeCon
North America 2022