EDGE APPLICATIONS: DEVELOPMENT OF STANDARDS SUPPORTING AN OPEN EDGE COMPUTING ECOSYSTEM

Michael McCool

Principal Engineer, Intel
W3C Web of Things WG Co-chair

OUTLINE

- **Motivation**
  - Use Cases
  - Definition and Goals
- **Possible Technical Approaches**
  - Discovery
  - Compute Service Offload
  - Orchestration Service Installation and Management

OUTLINE

- **Motivation**
  - Use Cases
  - Definition and Goals
- **Possible Technical Approaches**
  - Discovery
  - Compute Service Offload
  - Orchestration Service Installation and Management

USE CASES

Smart Retail

- Small business owners self-managing technology (1)
- Large retail franchises pushing applications out to employees' own devices (BYOD)

Smart City

- 40% of smart city use cases require multivendor solutions (2,3)
- Cities need to develop third-party app ecosystem to best provide value to citizens

(1) https://www.conexxus.org/
(2) https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world
(3) https://machinaresearch.com/news/smart-cities-could-waste-usd341-billion-by-2025-on-non-standardized-iot-deployments/

GOALS

Develop Open Edge Computing Ecosystem

- Define a general-purpose “compute utility”
- Allow devices to **offload compute** to “nearby” compute utility infrastructure
- Provide **IoT orchestration** services via distributed infrastructure
- Support secure, monetized, differentiated (e.g. accelerated) edge computing services
- Allow users to easily find and access edge computing services
- Simplify development and deployment of applications
- Support development of a third-party application ecosystem

TARGET CAPABILITIES AND THEIR REQUIREMENTS

Capability 1: Compute Offload

- Allow browser-based mobile computing access to accelerated compute utility
- Allow small IoT devices access to accelerated compute utility
- Allow client computing access to accelerated compute utility
- Compute utility may be on-board (device), local (edge), or remote (cloud)

➢ **Requirement:** Access to accelerated computing (GPU, FPGA, NN-ASIC, etc)

Capability 2: IoT Orchestration

- Install programmed orchestration function for derived IoT services

➢ **Requirement:** Access to local network and IoT devices

➢ **Requirement:** Persistent installation and event-driven execution

Other General Requirements

➢ **Privacy:** Trusted information and metadata management

➢ **Security:** Integrity, confidentiality, access control, authentication

➢ **Discovery:** Local and remote, devices and services, open but protected

➢ **Management:** Installation, cancellation, monitoring, payment

EDGE COMPUTING AND IOT ORCHESTRATION

Better Together!

- IoT Orchestration by itself is not compute-intensive
  … although some potential applications need compute-intensive services
- Edge Computing by itself lacks access to data and services

IoT orchestration + Edge Computing has *many* applications

- **Security:** motion sensor, camera, *person detection*
- **Inventory:** door open sensor, *product identification*
- **Logistics:** location tracking, 3D scanning, camera, *path planning*
- **Energy:** temp sensor, heater control, *person detection*, *machine learning*
- **Marketing:** door sensor, proximity sensor, camera, *sentiment analysis*
- **Cleaning:** robot vacuum cleaner, *obstacle classification, path planning*

OUTLINE

- Motivation
  - Use Cases
  - Definition and Goals
- **Possible Technical Approaches**
  - Discovery
  - Compute Service Offload
  - Orchestration Service Installation and Management

SUMMARY OF PROPOSED TECHNICAL STANDARDS STRATEGY

Extend PWAs, Service Workers, and Web Workers

- Web Workers extended to “Edge Workers”, supporting remote install on compute utilities, persistent lifetimes, event-driven execution, accelerated computing (e.g. via WebNN, TensorFlow.js, etc.), and to the local network for IoT orchestration
- PWAs/Service Workers extended to “Edge Apps”, supporting management lifecycle and remote “Edge Worker” components on compute utilities
- Use of WASM to package Edge App components offloaded to compute utilities.

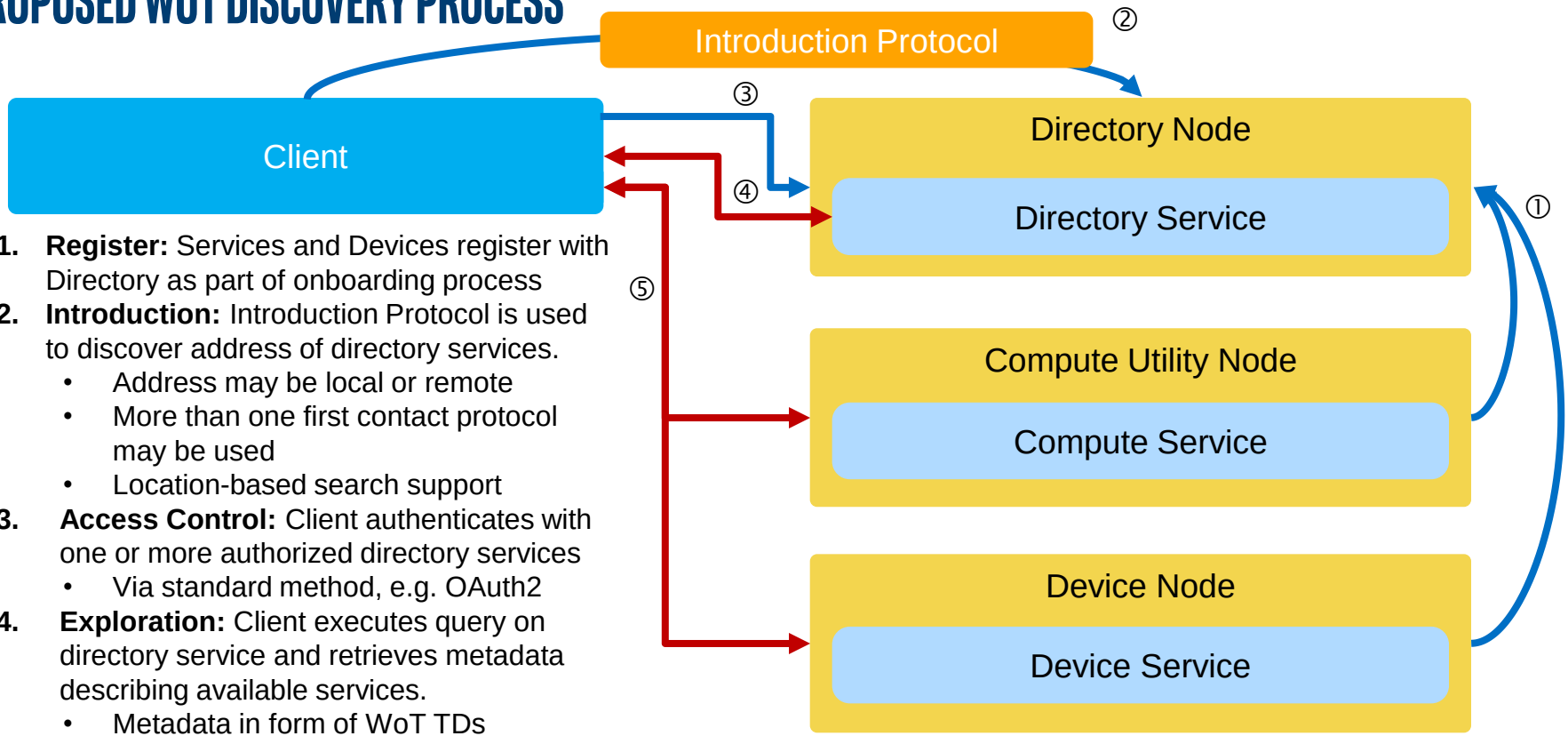
Extend Web of Things

- Extend WoT Discovery (WIP) to also apply to compute utilities
- Support IoT orchestration via WoT Scripting API (WIP) in Edge Workers

New Standards Development

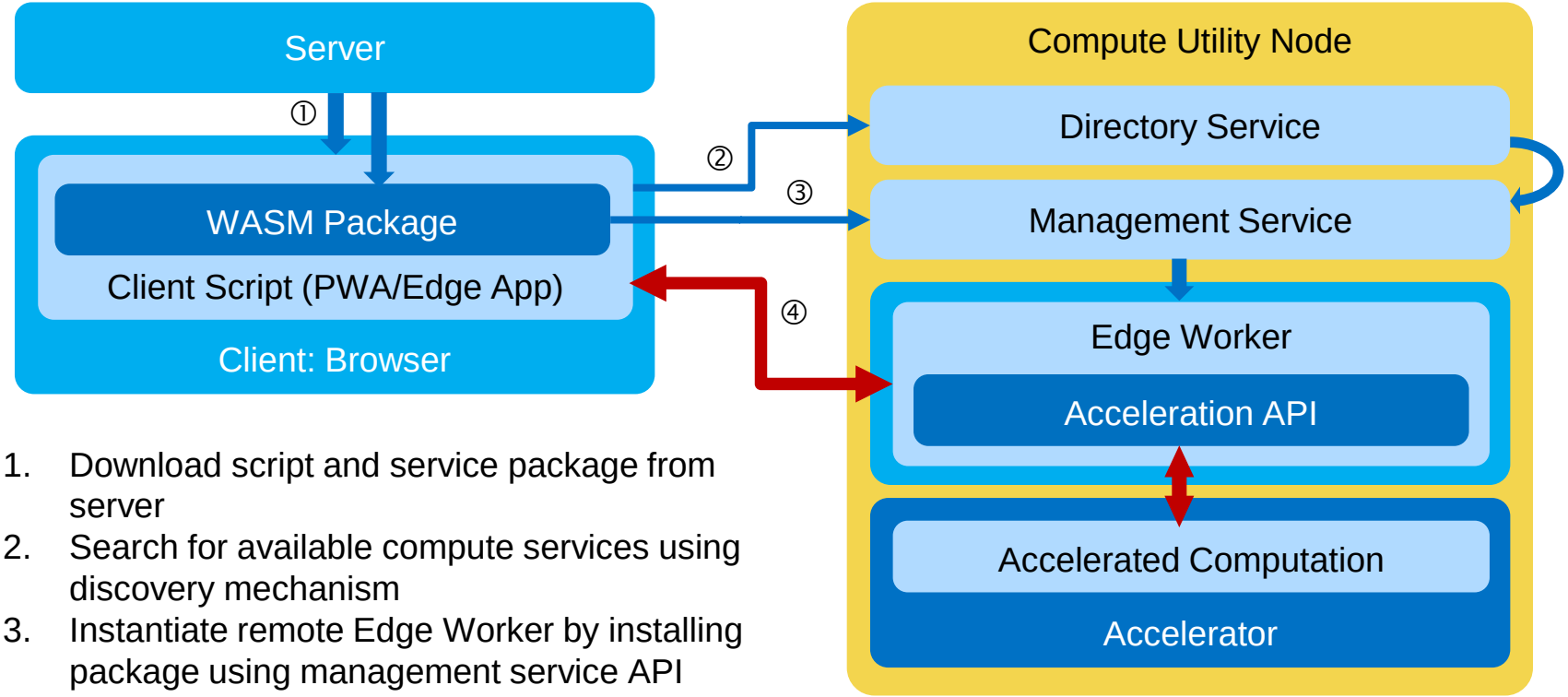
- Standardized Management API for compute utilities

PROPOSED WOT DISCOVERY PROCESS



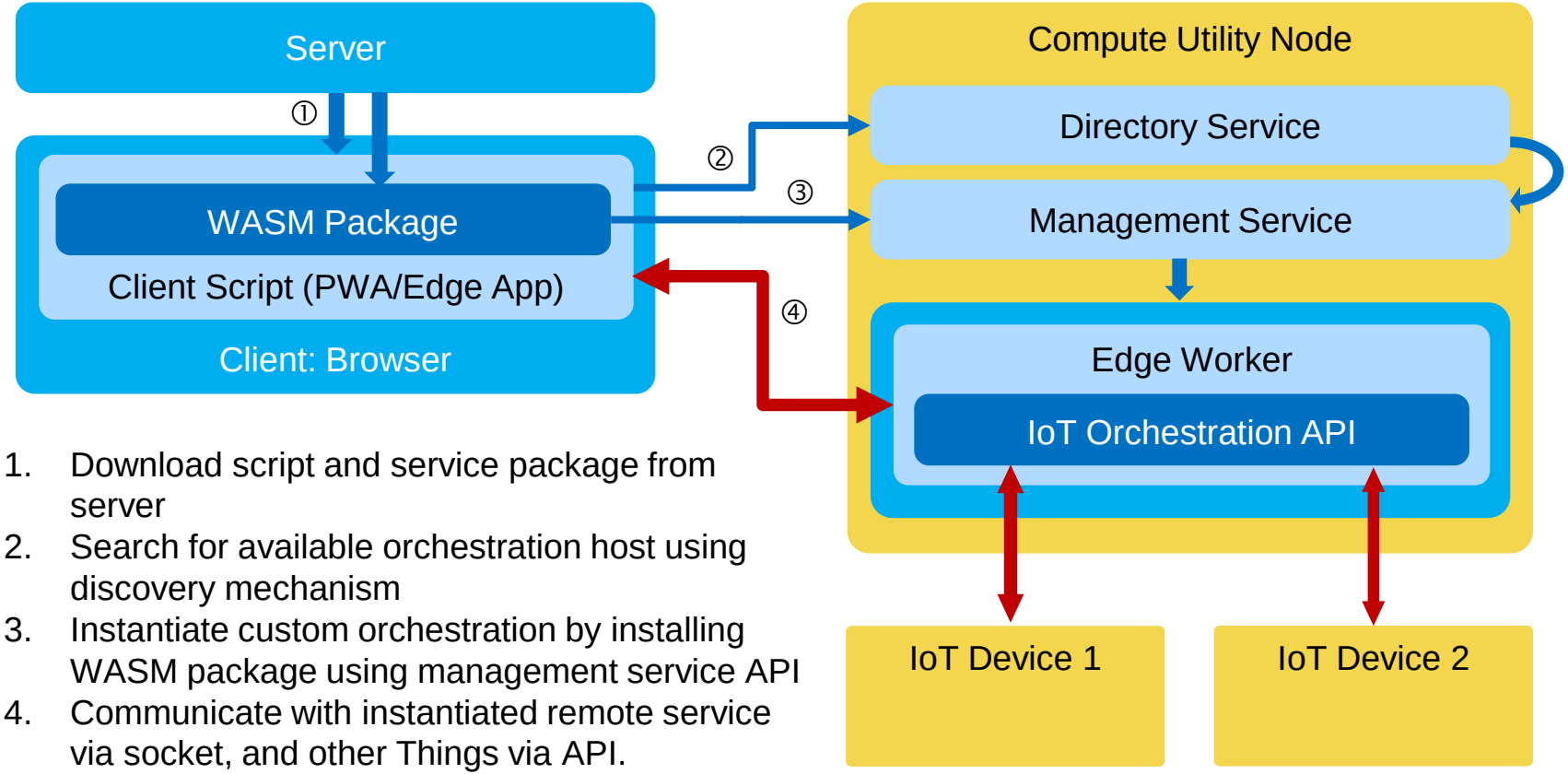
1. **Register:** Services and Devices register with Directory as part of onboarding process
2. **Introduction:** Introduction Protocol is used to discover address of directory services.
   - Address may be local or remote
   - More than one first contact protocol may be used
   - Location-based search support
3. **Access Control:** Client authenticates with one or more authorized directory services
   - Via standard method, e.g. OAuth2
4. **Exploration:** Client executes query on directory service and retrieves metadata describing available services.
   - Metadata in form of WoT TDs
   - Semantic search support possible
5. **Use:** Client accesses services.

EXAMPLE: COMPUTE SERVICE ACCESSED AS EXTENDED WEB WORKER



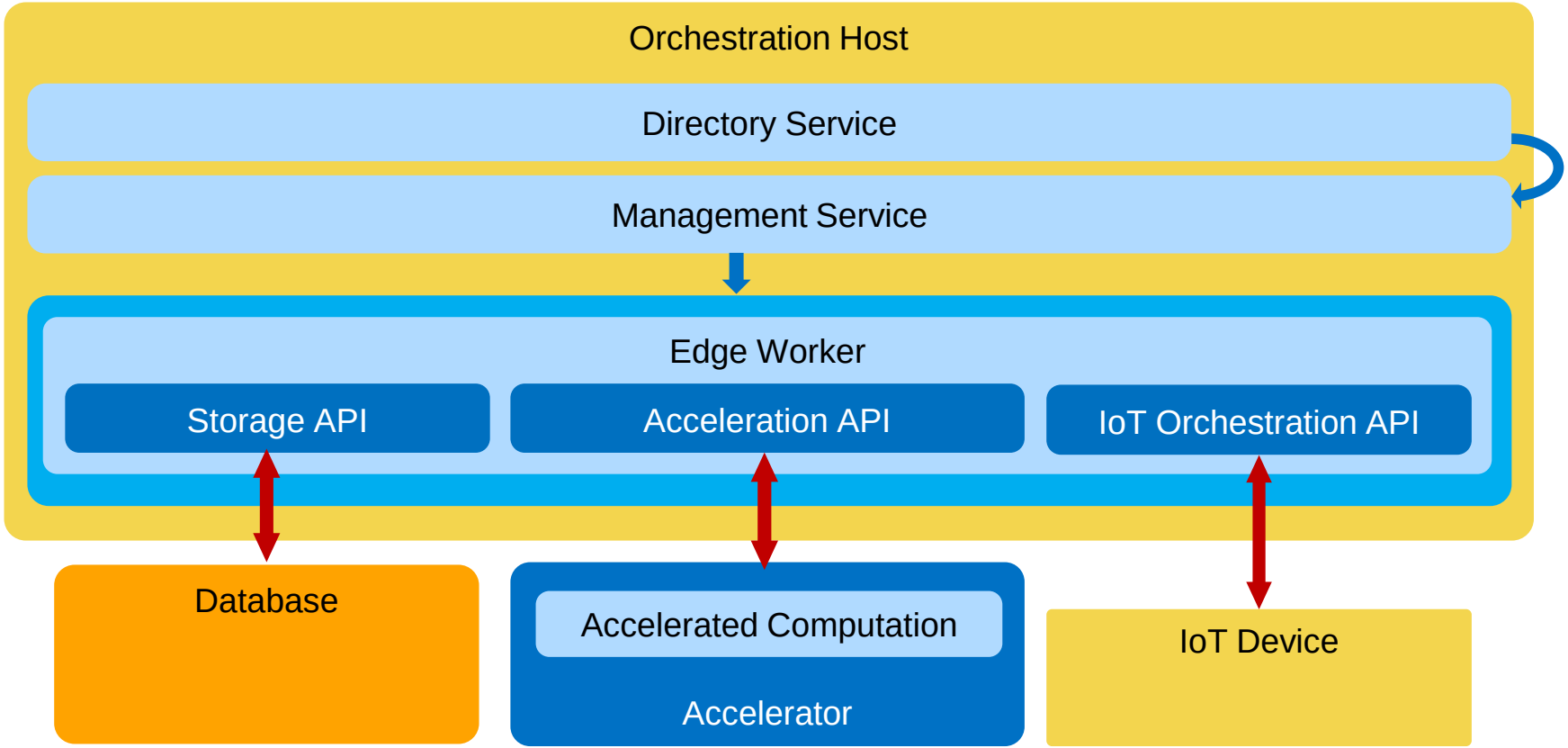
1. Download script and service package from server
2. Search for available compute services using discovery mechanism
3. Instantiate remote Edge Worker by installing package using management service API
4. Communicate with instantiated worker service via socket, and then to accelerator via API.

EXAMPLE: ORCHESTRATION SERVICE INSTALLED AS EXTENDED PROGRESSIVE WEB APPLICATION



1. Download script and service package from server
2. Search for available orchestration host using discovery mechanism
3. Instantiate custom orchestration by installing WASM package using management service API
4. Communicate with instantiated remote service via socket, and other Things via API.

MULTIPLE APIS ACCESSED FROM EDGE WORKERS



REQUIRED STANDARDS

1. **Discovery (network and maybe scripting API)**
- Find a compute node that can host a service (requirements-based search)
- Can be an extension/application of general IoT discovery process

2. **Management API (network and scripting) to Instantiate Services**
- API for a compute service that allows installation of a packaged service

3. **Packaging and Worker Management (e.g. WASM, containers, etc.)**
- Encapsulation of services that allows them to be installed in a sandboxed and isolated environment with all their dependencies and suitable (but controlled) access to other services

4. **APIs for Compute Acceleration (e.g. WebNN) and IoT Device Access (e.g. WoT)**
- Orchestration services need to discover and access other IoT devices
- Compute services need access to accelerated compute capabilities
- Discovery and installation of services should be possible from browser and web application contexts, e.g. as extension of PWAs and/or web workers