

Prototyping Distributed IoT Applications with WebRTC

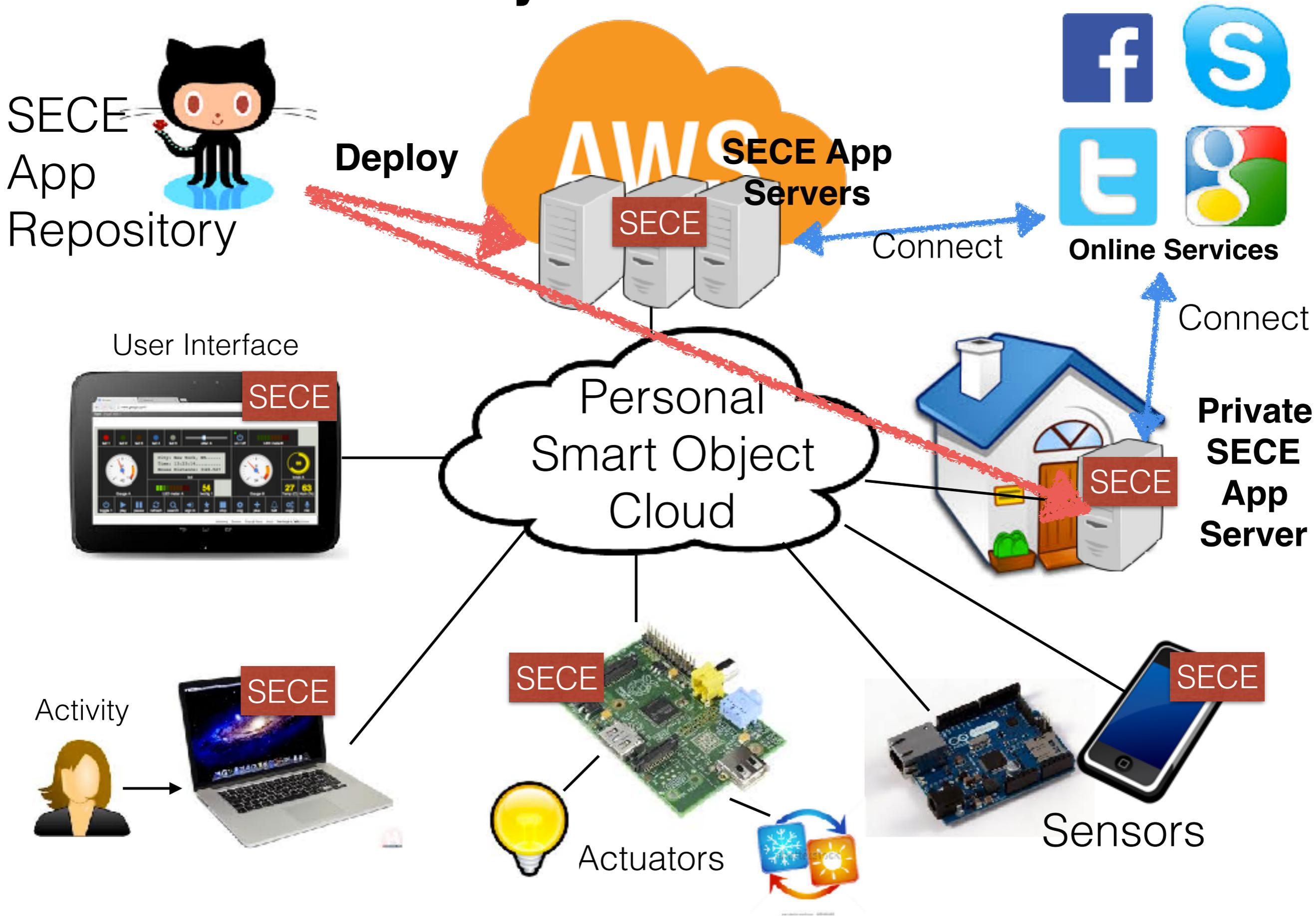
Jan Janak and Henning Schulzrinne

Internet Real-Time Laboratory
Columbia University

SECE: Sense Everything, Control Everything

- Research Project
- Distributed end-to-end programmable IoT applications
- Deploy IoT applications to end-devices
- Application model: computation follows data
- Benefits: security & privacy, scalability, resiliency
- Challenges: heterogeneity, concurrency, hardware

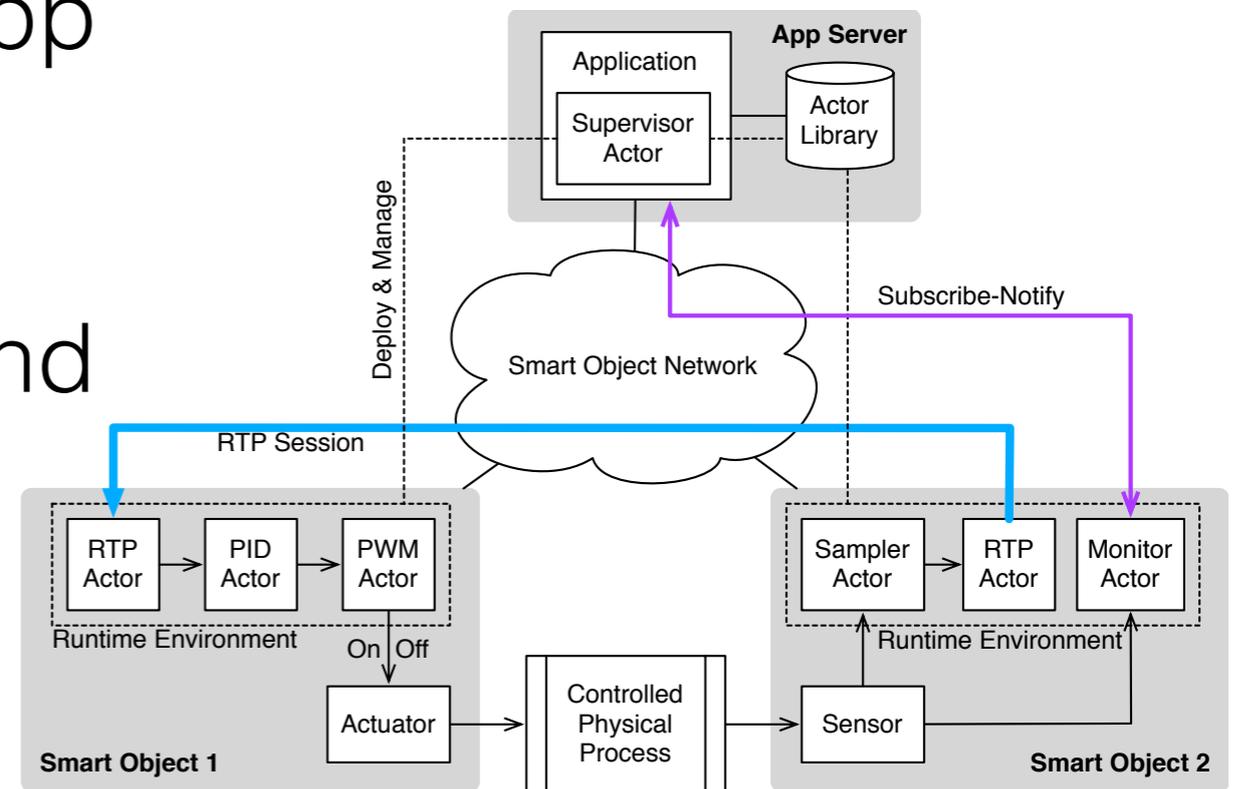
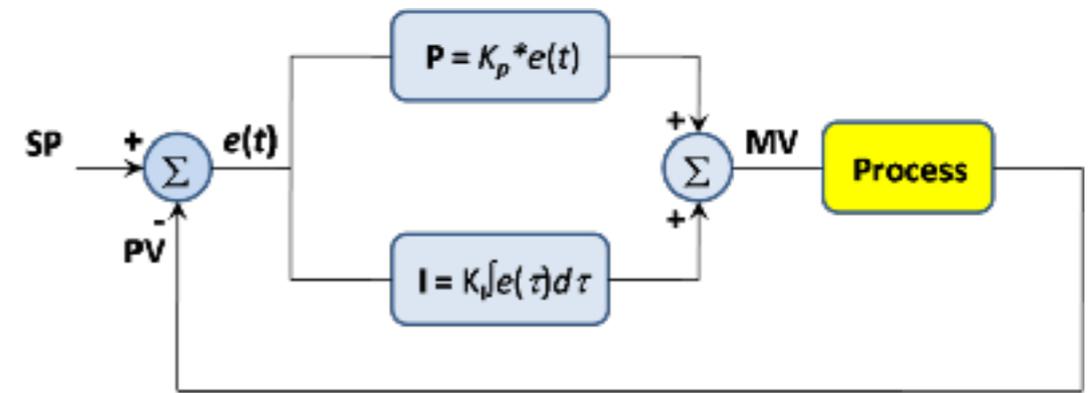
SECE System Architecture



Use Case: Internet PLC

Use of device programmability for control loops

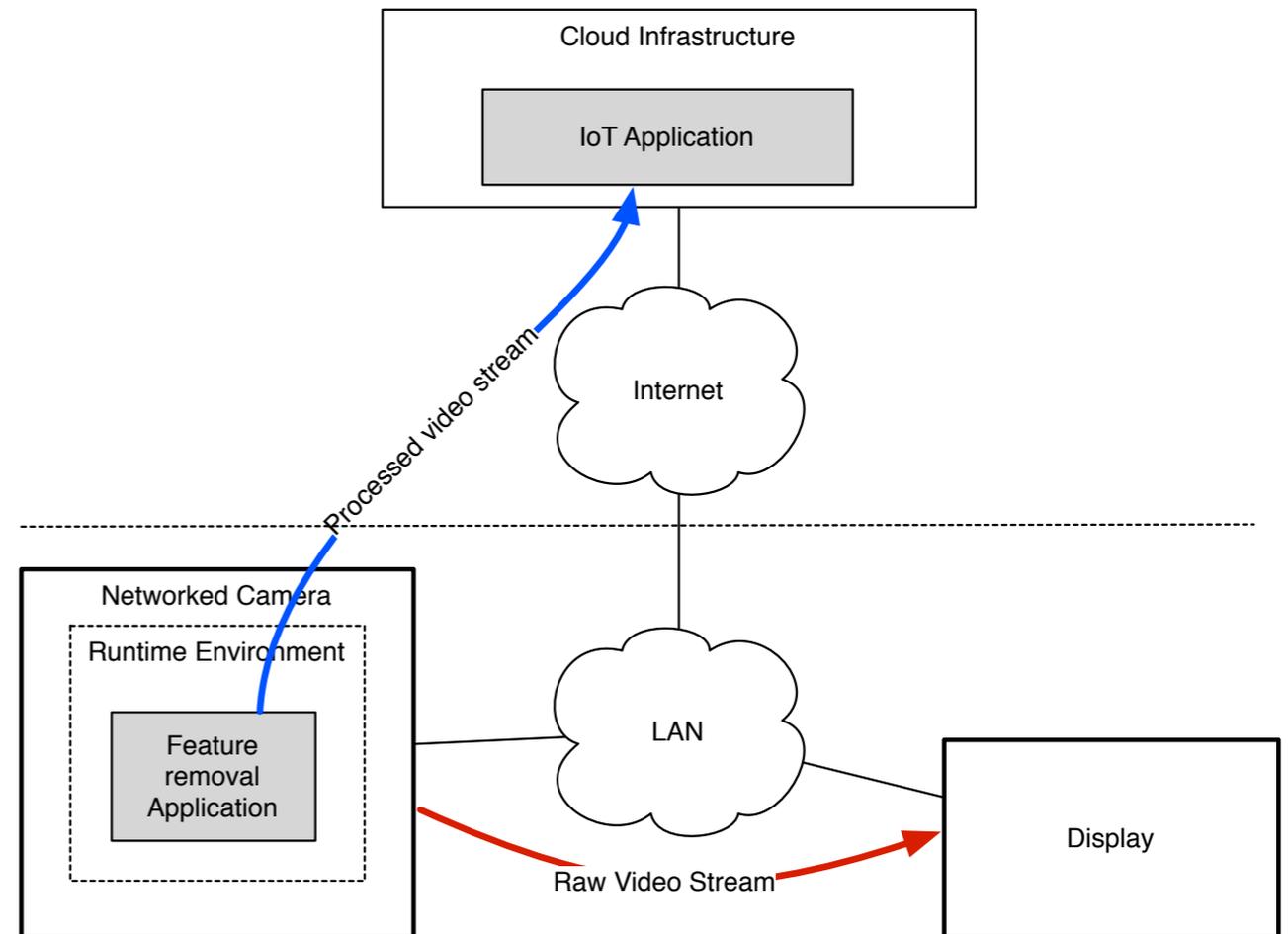
1. PID controller implemented on a programmable IoT device.
2. In form of a JavaScript / Lua application installed by SECE.
3. SECE requests persistent app installation on IoT device.
4. App continuously adjusts actuator based on history and sensor data.
5. App keeps running until explicitly uninstalled.



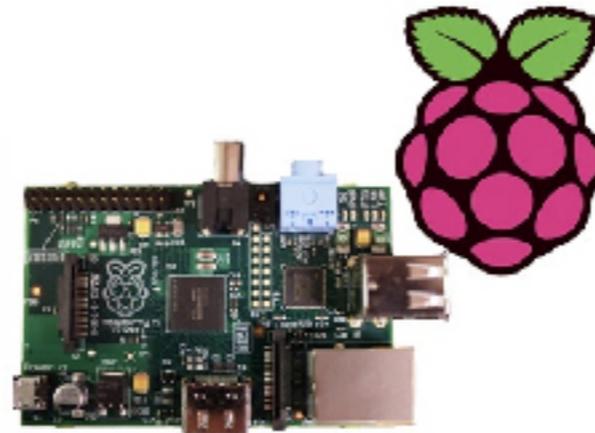
Use Case: Privacy Camera

Remove privacy sensitive features from video stream

1. Authorization required to obtain raw video from camera (e.g., on the same LAN)
2. SECE installs a feature removal app on the camera
3. App removes privacy sensitive features (B&W, crop, binary)
4. App streams processed data to cloud for further processing



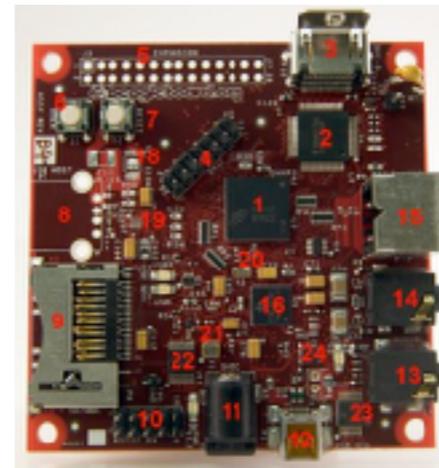
Challenges: Device Heterogeneity



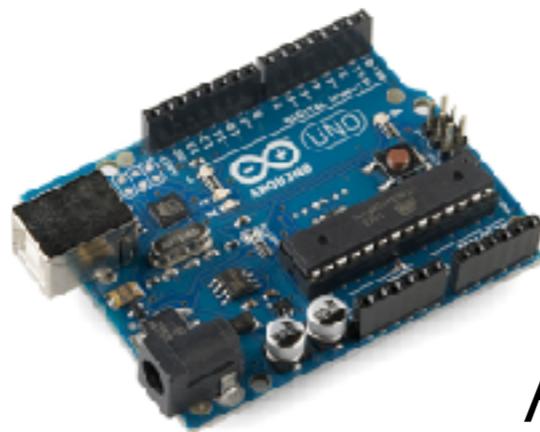
BeagleBoard



Phidgets



Arduino



WebRTC & Distributed IoT Apps

- Enable direct Device-to-Device communication
 - Closed loops, latency, privacy
- Make prototyping without hardware possible
- First class support for browser-enabled devices
 - UI, programmability, storage, communication
- Component design

Web-based User Interface

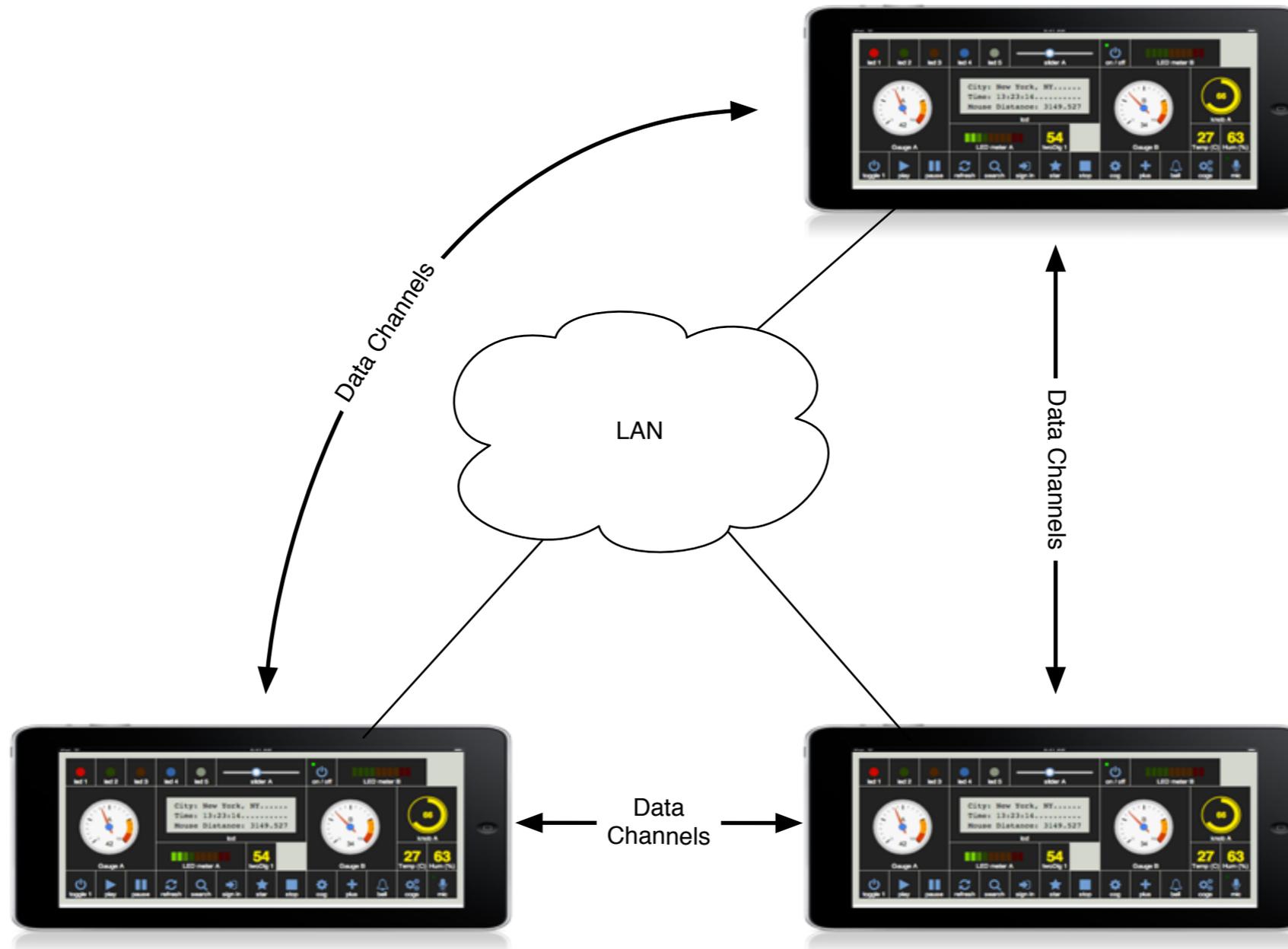


Web-based User Interface

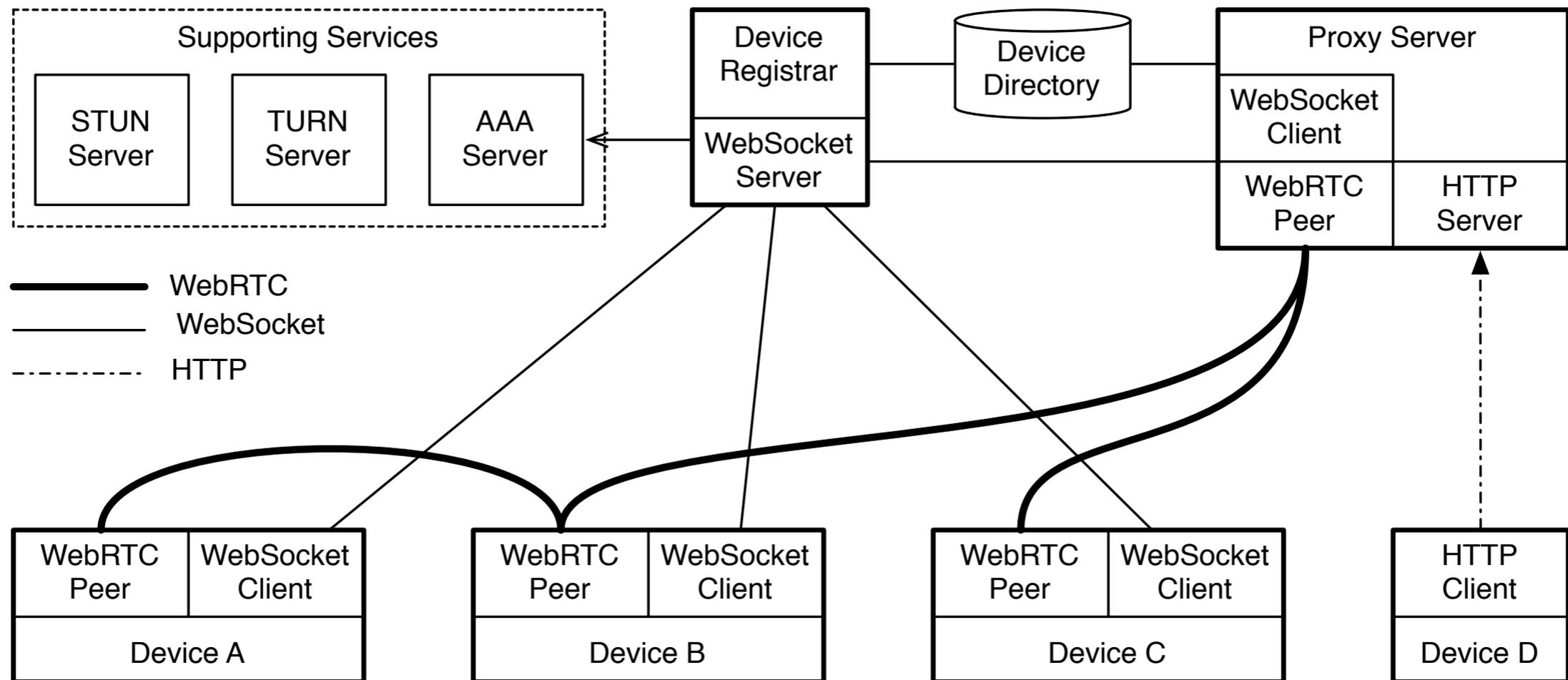
- Use HTML5 and WebRTC for communication
- Virtual widgets implement inputs and outputs
- Gauges, LEDs, Buttons, A/V inputs/outputs
- Runs on a tablet with a recent browser
- Modes of operations
 - MVC: multiple views of a panel (synchronized) (requires node.js server)
 - Only-one instance (fails to open second time)
 - Bound to a specific device



Direct Device-to-Device Communication



Network Architecture (for WebRTC)



IoT Device Emulation

- Emulate simpler IoT devices in JavaScript
 - Arduino, Teensy, ZigBee, and similar
 - Run program for the device via Emscripten
 - Connect I/O ports to virtual HTML5 ports
- Inspiration: Internet Arcade
- Why JavaScript?
 - Available everywhere (tablets, laptops, Raspberry PI, Node)
 - Interactive (UI capabilities)

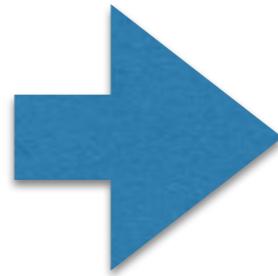
Inspiration: Internet Arcade

- An effort to preserve computer history
- 900 classic games in your browser
- Emulates arcade console hardware in JavaScript
- Runs unmodified binary images of games



<http://www.theverge.com/2014/11/2/7147505/the-internet-arcade-puts-900-classic-games-right-in-your-web-browser>

Internet Arcade



MAME: SG-1000 [sg1000]

https://archive.org/details/Space_Invaders_1985_Sega_Taito

INTERNET ARCHIVE

ABOUT CONTACT BLOG PROJECTS HELP DONATE TERMS JOBS VOLUNTEER PEOPLE

SCORE (1) HI-SCORE SCORE (2)

Space Invaders

Space Invaders

by Segs - Taito

Published 1985

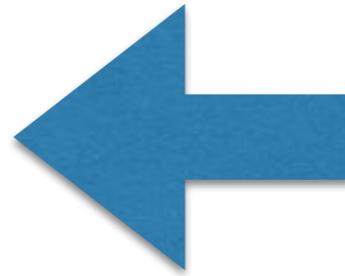
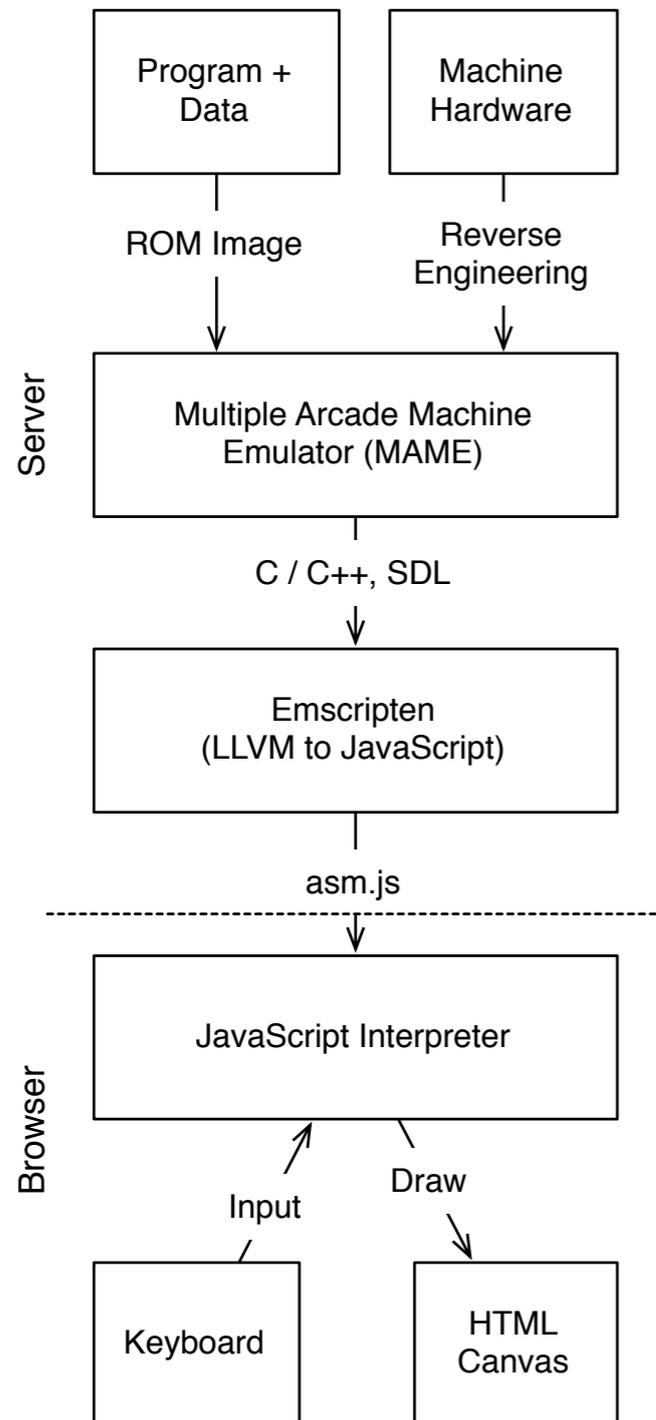
Space Invaders (スペースインベーダー) is a 1978 milestone arcade game developed by Taito. The player controls a gun which can move left or right, and needs to destroy waves of aliens (or "invaders") before they reach the ground. Space Invaders was brought to a variety of home video game consoles and computers, including the SG-1000 in 1985.

Surprisingly the SG-1000 port of the game was one of the most accurate of its time, surpassing the NES and MSX versions which were the console's main competitors. Though it adds more color (the arcade version was simply black and white, with translucent overlays to mimic color), the SG-1000 version retains all the original graphics, whereas other ports (especially for Atari systems) re-drew the invaders

In Collection
**Console Library:
Sega SG-1000**
AND 1 MORE

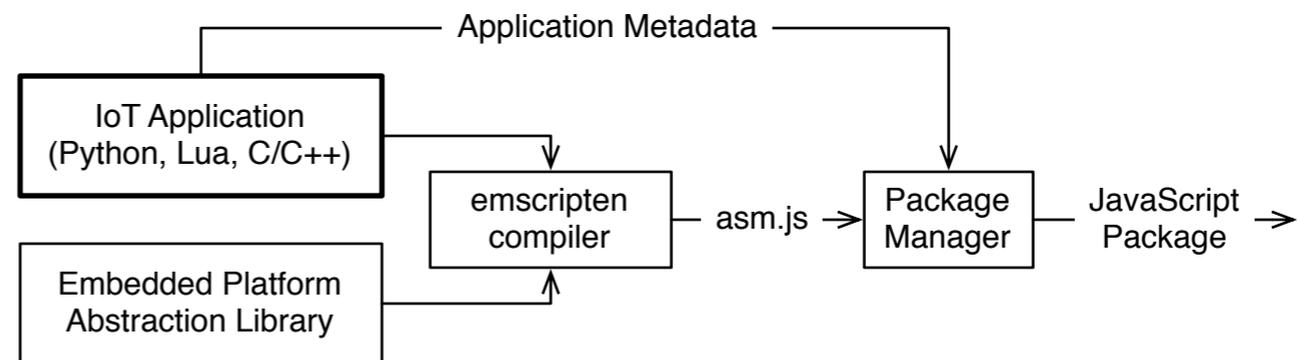
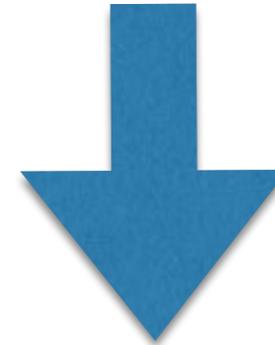
JavaScript IoT Device Emulation

Work in Progress

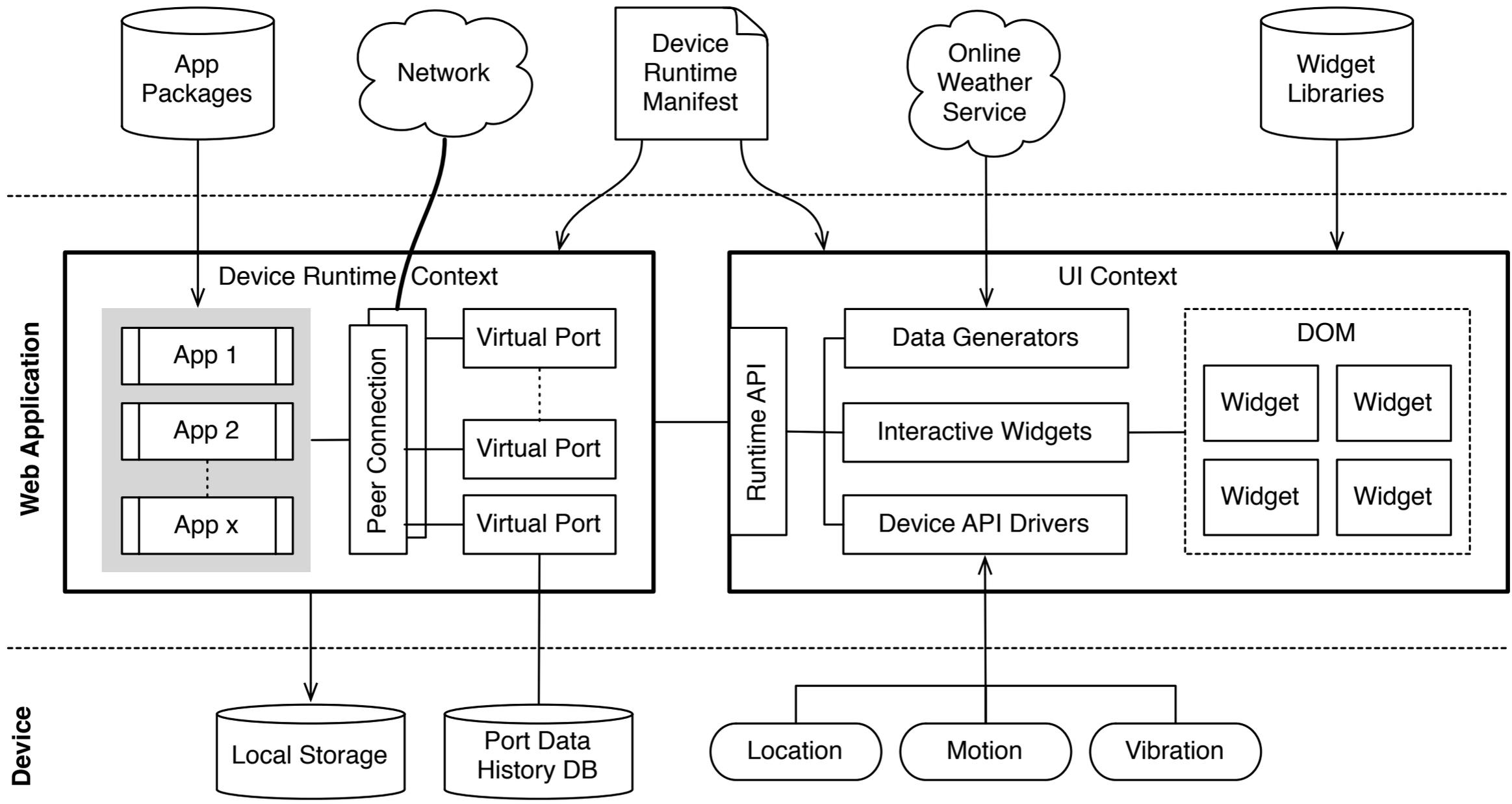


Internet Arcade

IoT Devices



SECE JS Framework



Summary

- End-to-end programmable IoT applications
 - Privacy, security, latency, resiliency benefits
 - Hard to design, prototype, and test
- HTML5 + WebRTC + JavaScript to the rescue
 - First-class virtual ports modeled after physical ports
 - WebRTC for direct communication between ports
 - IoT device emulation in JavaScript feasible