The Internet of Things in a Laptop:
Rapid Prototyping for IoT Applications with Digibox

Silvery Fu, Hong Zhang, Sylvia Ratnasamy, Ion Stoica
UC Berkeley
**IoT: devices and apps**

~7.74 billion connected IoT **devices**

In homes, offices, retail locations, commercial buildings..

New **applications**: smart spaces, logistics, agriculture, urban sensing..

*Picture courtesy of Comfy: https://comfyapp.com/*
**IoT:** devices and apps

- **E.g. Smart building**

  - **Physical infrastructure**

  - **App framework**
    - **App logic**
      - **Driver**
      - **Device**

- **IoT:** devices and apps
  - E.g. Smart building
  - Physical infrastructure
IOT: devices and apps

Setting up the testbed: a bottleneck in prototyping e.g., research, class, demo

Time, Cost

Physical infrastructure

Device
Mininet

Bob Lantz, Brandon Heller, and Nick McKeown, HotNets 2010

Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine (VM, cloud or native), in seconds, with a single command:

```
> sudo mn
```

Because you can easily interact with your network using the Mininet CLI (and API), customize it, share it with others, or deploy it on real hardware, Mininet is useful for development, teaching, and research.

Mininet is also a great way to develop, share, and experiment with Software-Defined Networking (SDN) systems using OpenFlow and P4.

Mininet is actively developed and supported, and is released under a permissive BSD Open Source license. We encourage you to contribute code, bug reports/fixes, documentation, and anything else that can improve the system!

---

### Get Started
- **Download** a Mininet VM, do the walkthrough and run the OpenFlow tutorial.

### Support
- Read the FAQ, read the documentation, and join our mailing list, mininet-discuss.

### Contribute
- File a bug, download the source, or submit a pull request - all on GitHub.
IoT: Prototyping Env.

Real-world testbed
Human interaction
Correlated behavior

Today: device simulator
H/W capability
Individual device behavior
e.g. events, messages

Ease-of-setup

Fidelity

Real-world testbed

Device simulator

?
IoT: Prototyping Env.

Fidelity

Real-world testbed

Ease-of-setup

Device simulator

Real-world testbed

Human interaction
Correlated behavior
Easy to reproduce
Easy to (re)use
Scalable

Individual device behavior
e.g. events, messages

Today: Device simulator

Digibox

Easy to reproduce
Easy to (re)use
Scalable
IoT: Prototyping Env.

Key insight: Device-centric → Scene-centric

Real-world testbed

Digibox

Device simulator

Scene

Today: device simulator
H/W capability
Individual device behavior
e.g. events, messages

Fidelity

Ease-of-setup
Digibox Walkthrough w/Demo
Digibox

Design
Mock
Scene
Attach

Property
Interactive
Ensemble support
Reproducible
Customizable
Scalable
**Goal: Interactivity**

- **CLI:** `digi` (alias: `dbox`)
- **Command:** `pull/run`, `check/edit/query`

In the demo:
- **Mockup device**
- **Lamp**

**Digibox**

- Design
- Mock
- Scene
- Attach
- **Property**
- Interactive
- Ensemble support
- Reproducible
- Customizable
- Scalable
Goal: Interactivity

Each mock has a **model** and a **simulator**

**Model**: declarative interface, intent & status

**Simulator**: a piece of Python code

.. Simulator subscribes to intent changes and update the status on the model

Each mock runs in its own microservice ("digi")

Isolation: separate dev./deployment lifecycle
**Goal: Ensemble Support**

In the demo:
- Scene
  - Room
    - Sensor: "ceiling"
    - Sensor: "desk"

**Command:**
- `attach`

**Digibox**
- Design
  - Mock
  - Scene
  - Attach
- Property
  - Interactive
- Ensemble support
  - Reproducible
  - Customizable
  - Scalable
Goal: Ensemble Support

Scene: same components as a mock (model & simulator)

Attach(M, Sc):
allows Sc.sim to write to the mock's model M
Goal: Ensemble Support

Scene composition

num_human: 1
human_presence: true

Building
human_presence: false

Office

Lamp
detected: false

Ceiling
detected: true

Desk

Lamp

detected: false

num_human: 1
human_presence: true

Building
human_presence: false

Office

Lamp
detected: false

Ceiling
detected: true

Desk

Lamp

detected: false
Goal: Share and Reproduce

In the demo:
Take a snapshot of mocks/scenes

Command:
commit
push
See the paper for details:

**Programming Mocks and Scenes**
- Creating, programming, and configuring digis
- Integrating mocks with app frameworks

**Implementation and Scalability**
- Kubernetes-based runtime
- Scaling from a laptop to machine cluster in cloud

**Sharing and Reproducing Scenes**
- Managing mocks and scenes with Git/GitOps
- Logging and replaying traces
Prototyping Workflow

App framework

1. Write app
2. Write scene
3. Run scene
4. Run app
5. Debug

Digibox

Pull scene
Run scene
Replay logs

Design
Mock
Scene
Attach

Property
Interactive
Ensemble support
Reproducible
Customizable
Scalable
Goal: Accelerate IoT app prototyping (sim./testing)

Interactive, ensemble behaviors, reproducible, scalable

Digibox
Mock, Scene + Attach

digi.dev/digi
Looking ahead

"Net Apps"
Network Apps
Network Apps

Mininet

Interacting with network devices
→ the physical world

Open Challenges

Digibox

Prototyping environment:
High-fidelity simulation
Efficient, large-scale simulation
Supporting new applications

Prototyping environment:
High-fidelity simulation
Efficient, large-scale simulation
Supporting new applications
Goal: Accelerate IoT app prototyping (sim./testing)

Interactive, ensemble behaviors, reproducible, scalable

Digibox

Mock, Scene + Attach

Thank you!

digi.dev/digi