Context-aware Edge Process Management for Mobile Thing-to-Fog Environment

Jakob Mass, Chii Chang and Satish Srirama

Institute of Computer Science, University of Tartu

September 26, 2018
About me

PhD student @

Mobile & Cloud Computing Lab

Institute of Computer Science, University of Tartu
Estonia
Outline

- Background:
  - Service-oriented Internet of Things
  - Edge Process Management

- Research Question - Mobility

- System Architecture Overview

- Experimental Results

- Conclusion
Services-oriented IoT
Workflow Management Systems

- Model group of devices as composite service - a workflow
- WfMS orchestrate, manage and execute these workflows
- Standards such as BPMN 2.0 for defining the workflow (process)
- Traditionally hosted in remote centralised server

- Decision-making, messaging, event handling
Edge Process Management (EPM)

EPM - Emphasis on decentralised processes

- Distribute tasks to the edge network
- Reduce server-side bandwidth
- Reduce client latency

Use cases & application

- Remote health care, Smart traffic control, Disaster Recovery
- **Fog Computing**: a mobile node distributes computational task to the proximal fog server
In case a mobile node needs to execute a task involving nearby wireless fog servers, the result is affected by:

- Fog server hardware configuration
- Fog server workload
- Signal strength
- **Movement trajectory** - Internet of Mobile Things (IoMT)
Mobility-related challenges

Executing tasks while signal area is encountered briefly:

- task fails
- task re-executed locally or at next fog server
- resources wasted

Execution with weak signal:

- poor performance
- delays
Mobility-related challenges

Executing tasks while signal area is encountered briefly:

- task fails
- task re-executed locally or at next fog server
- resources wasted

Execution with weak signal:

- poor performance
- delays

Research Question

How can EPM systems support adaptive task execution in the Mobile Thing-to-Fog environment?
Proposed system

Primary goal - managing workflow task execution schedule based on runtime factors

- IoMT Server
  - IoMT server device
  - co-located sensor devices
- Fog Server
- Cloud Management Server
Fog Server

- Advertises system status to nearby clients
- Deployment & Execution Platform
  - Tasks (incl. input data) packaged as *App Deployment Packages (ADM)*
  - Results returned as a *Result package*. 
IoMT server

IoMT Server

Cloud-to-Thing Agent

Thing-to-Fog Agent

Workflow Execution Engine

Sensory Data Manager

Embedded Sensor Adaptor

LoWPAN Client

Fog Server
IoMT - Thing-to-Fog Agent

Performs continuous fog server discovery in the background.

Discovery info:
- Hardware specification
- Current system workload
- Available application deployment platform

_Fog App Agent_ provides means to distribute tasks to the _App Platform_ on the server.
Workflow Manager

- Maintains WF model DB
- Interfaces with OS components
- Assigns Task Managers to workflow tasks
Task Manager

- controls execution of individual WF tasks
- execution schedule based on Context Manager
Context Manager

- Interprets sensory data & discovery data
- Includes user movement trajectory
- In this work, we focus on signal strength and mobility
WF models sent to IoMT server from Cloud
Sensor Adapters provide interfaces to WF Engine
LoWPAN Client includes support for auxiliary wireless sensor devices
Task Scheduling

As a formal basis for dynamic task execution schedule, we derived a Time Petri Nets based Scheme.

- We define Adaptive Time Petri Nets (ATPN)
- Execution is bound by a contextual earliest firing time and latest firing time, which may change values

![Diagram](a) Time Petri Net

![Diagram](b) Adaptive Time Petri Net

**Figure:** Modelling of the system with different scheduling approaches.
Experiments

Demonstrate the effect of scheduling decisions with real devices
- IoMT server: Android smartphone
- Fog server: PC + WiFi router
- Task: Connect & download payload
- Movement: hallway walking

Compare:
- baseline - Signal Strength Threshold (SST) approach
- mobility-based scheduling approach (MOBI)
Experiments - single instance comparison

Mobility-based scheduling (MOBI) against Signal Strength Threshold (SST) scheduling.

(a) SST: -85dB.  
(b) SST: -65dB.  
(c) MOBI.

Figure: Timeseries comparison.
Experiments

(a) Total time of task.  
(b) Average RSSI during download.

Figure: Performance with different scheduling configurations.
Discussion

- SST approach needs manual tuning. Tighter constraints generally improve performance, but the danger of over-constrainting exists.
- MOBI approach both improves performance while having more stable signal throughout task.
- On the other hand, MOBI is dependent on quality of mobility modelling and prediction.
Conclusion

- Presented Architecture for Edge Process Management
  - Process-based Fog Task distribution
  - ATPN modelling for task schedules

- Experimented task scheduling with devices
  - Simple threshold-based approach can be outperformed
  - However, influenced by mobility prediction accuracy

- Future work
  - Mobility prediction algorithms
  - Consider other context like system load
  - Integrate with existing BPMN tools (e.g. Camunda)
Thank you for listening!