IoT Design Patterns: Computational Constructs to Design, Build and Engineer Edge Applications

Soheil Qanbari*, Samim Pezeshki†, Rozita Raisi †, Samira Mahdizadeh*, Rabee Rahimzadeh†, Negar Behinaein†, Fada Mahmoudi†, Shiva Ayoubzadeh †, Parham Fazlali†, Keyvan Roshani†, Azalia Yaghini†, Mozhdeh Amiri†, Ashkan Farivarmoheb†, Arash Zamani†, and Schahram Dustdar*

*Distributed Systems Group, Vienna University of Technology, Vienna, Austria
Introduction

- There are huge opportunities but considerable challenges in designing IoT applications
  - Provisioning of ultra-low power operation and system design using modular, composable components to smart automation
  - Sensor instrumentation requires an efficient stream data processing
- Propose patterns, which aid system architects in modeling and building context-tailored IoT applications
Pattern Language Conventions

- Pattern language is intended to describe the solution in a way that is easy to digest
- All IoT patterns comprise following semantics
  - Pattern Name
  - Problem
  - Context
  - Motivation Forces
  - Solution Details
  - Sketch
IoT Design Patterns

- Connected devices may use different types of networks and various connectivity patterns
  - Consider design on many different layers
- Automated provisioning, deployment and configuration management of the behavior of edge applications disclosed herein pertain to governance patterns
  - Govern all aspects of edge applications including their provisioning and deployment mechanisms
Edge Provision Pattern

- **Problem**
  - How to ensure all of edge devices are started with a reliable baseline environment, as needed?
  - How to provision all the devices automatically all at once?

- **Context**
  - IoT devices are usually scattered geographically, sometimes hard to reach and large in number
  - We must be able to reconfigure devices or provision new ones in an efficient way and have pre-configured nodes
Edge Provision Pattern

- Motivation Forces
  - Replace your technology stack entirely and provision a new environment remotely
  - Add new devices and provision their runtime environment and applications quickly

- Solution Details
  - Container-based virtualization for provisioning
  - Have pre-configured environment with required applications installed
  - Docker images utilize a layered and versioned file system
Edge Provisioning Pattern Sketch

Provisioning Server

- Container Builder
- Containerizer APIs
- Isolator APIs

Developer

- Container file

Container Registry

- C1
- C2
- D1
- D2
- D3
- .....
Edge Code Deployment Pattern

- **Problem**
  - How to deploy code to many IoT devices automatically, quickly and safely, and configure them without being concerned about the long process?

- **Context**
  - Deploy the updated code to remote IoT devices quickly. This grants distributing functionality between devices
  - Re-configure the application’s environment
Edge Code Deployment Pattern

- **Motivation Forces**
  - Update the text or graphical features frequently or change the duration of ad display
  - It is best to only deliver the changes
  - The tools for deploying the code to devices should be transparent to the developers

- **Solution Details**
  - Version control systems
  - The deployment pipeline is started with each commit, and changes in the source code are published to all edge devices
Edge Code Deployment Pattern Sketch

- **Developer**
  - 

- **Version Control**
  - source code
  - env config
  - app config

- **Artifact Repository**
  - binaries
  - test reports
  - metadata

- **Provisioning Server**
  - Container Builder
  - Container Registry

- **Acceptance Tests**
  - commit tests
  - smoke test
  - code analysis

- **Edge Layer**
  - deploy (push/pull) image

- **Build Code to Binary**
  - Container file (spec, binary, config)

- **Unit Analysis/Test Results**
Edge Orchestration Pattern

Problem
- How to orchestrate IoT devices remotely?
- How can edge cluster nodes discover services?

Context
- The cluster manages nodes to check their health state, their services state to reconfigure them.
- Run services in the cluster on certain nodes and enable them to discover the services they need and re-configure themselves accordingly.
- Edge nodes should be able to advertise services they provide.
Edge Orchestration Pattern

- Motivation Forces
  - We need a declarative way to deploy home automation system without configuring and installing each component separately on every device

- Solution Details
  - Containers’ compose-oriented technology enables us to deploy composite applications
  - Service discovery mechanisms can be leveraged by nodes to find each other
  - [key,value] stores
Edge Orchestration Pattern Sketch
Edge Diameter of Things (DoT) Pattern

- **Problem**
  - How to monitor and meter the actual usage of IoT deployment units in real-time in order to monetize them?
  - How the IoT composite application resource usage can be charged against a specific user balance?

- **Context**
  - The need for defining some metrics for service and resource usage, which in turn, can be used to measure the consumption of the service and to price it
Motivation Forces

- Measure the rate of actual resource and service utilization, as near real-time as possible

Solution Details

- Define a specific agreement called metering plan, offered by the provider and accepted by the client
- Parses the plan and calculates the U3 (Used Unit Update) rate for each constituent service in the plan.
Edge Diameter of Things Pattern Sketch
Conclusion

- Defined four design patterns enabling IoT architects to construct edge applications
- They will focus more on patterns to be used for elasticity, resiliency and Software Defined Networking (SDN) patterns for edge computing