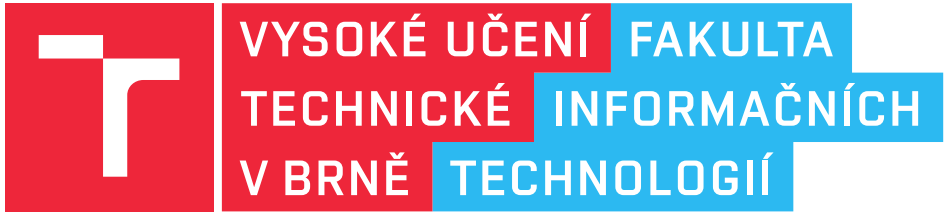
Rekonfigurovatelné IoT systémy

Dominik Václavík

Vedoucí: doc. Ing. Vladimír Janoušek, Ph.D.



27. ledna 2026

- Vývoj embedded IoT aplikací je složitý a obtížně aktualizovatelný
- Změna logiky často vyžaduje plný firmware update
- Cílem je flexibilní, event-driven, low-code systém pro ESP32

Klíčové vlastnosti:

- Dynamická konfigurace bez reflashe
- Logika pomocí nodů
- Lua, Micropython nebo nativní kód
- Wi-Fi, Zigbee, Thread/Matter
- Vlastní TCP Protokol, MQTT

High-Level Architecture

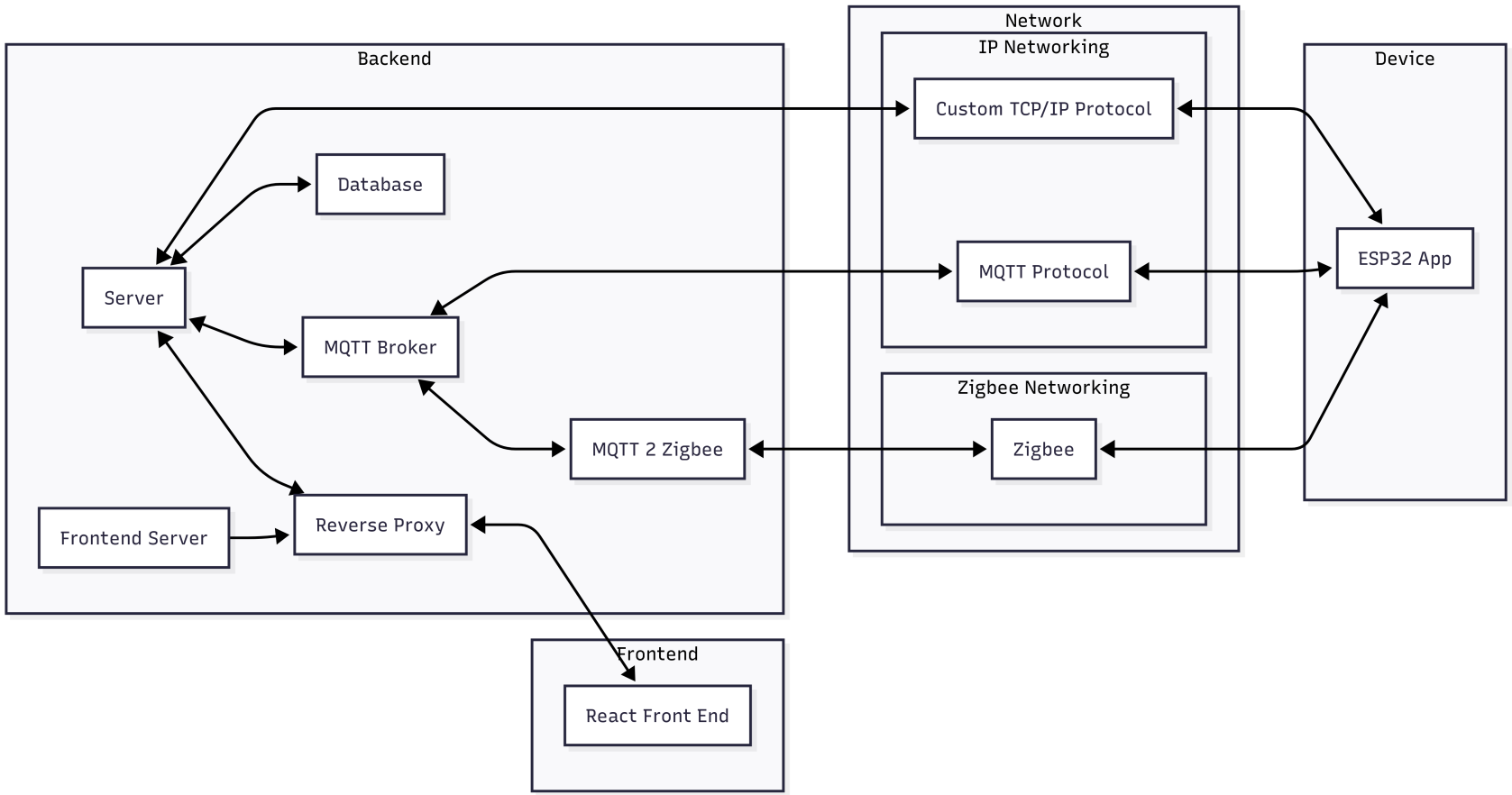
Backend / Frontend

- React frontend + React Flow (Javascript)
- Konfigurační server (Python)
- MQTT broker + Zigbee2MQTT
- Databáze

Zařízení

- ESP32 Firmware (C/C++, FreeRTOS)
- TCP / MQTT / Zigbee

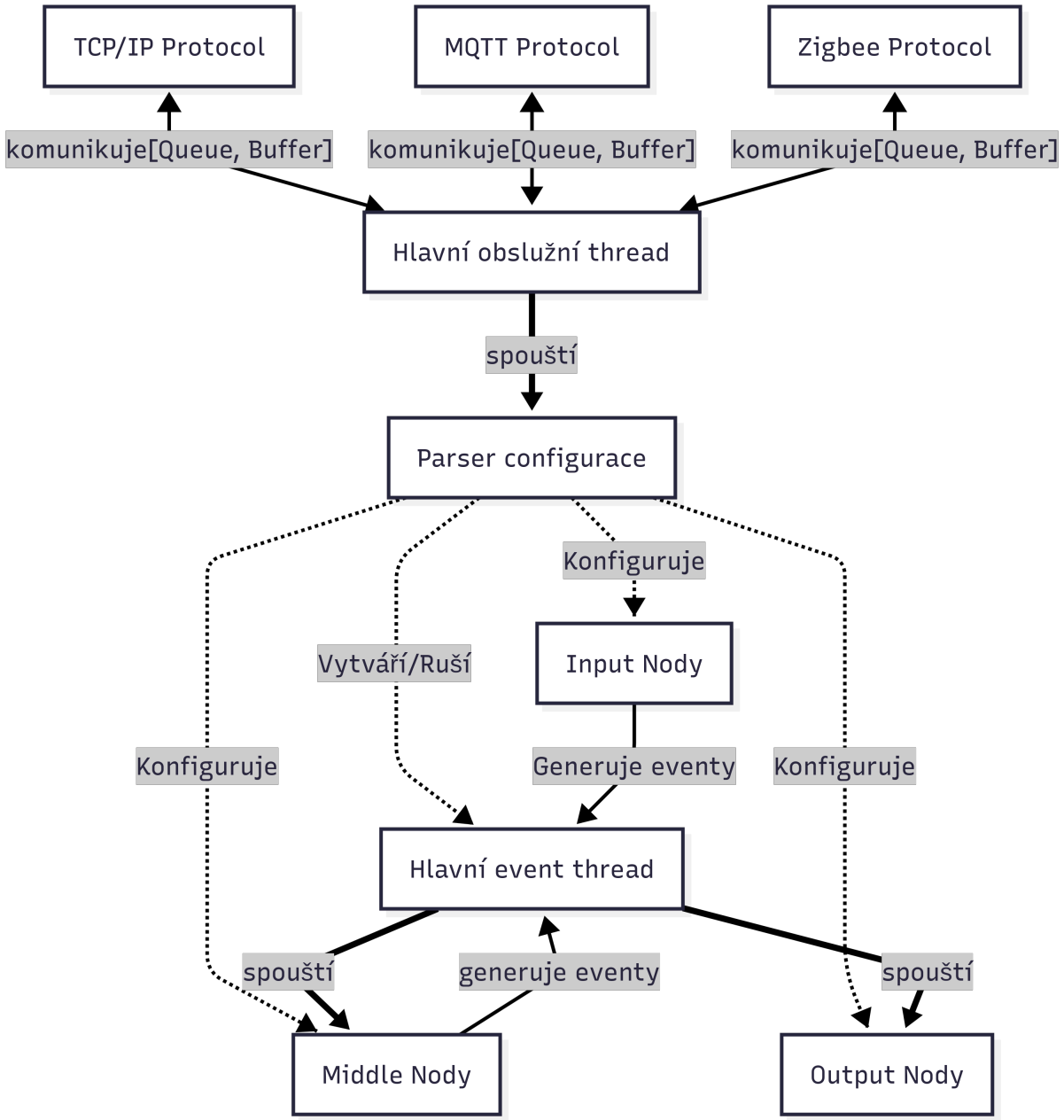
High-Level Architecture



FreeRTOS procesy / FreeRTOS tasky

- **Protokolové procesy / Protokolové tasky**
  - Komunikuje s hlavním procesem pomocí Queues
  - TCP Protokol, MQTT, Zigbee
- **Hlavní Proces**
  - Reaguje na zprávy od komunikačních protokolů
  - Parsuje příchozí konfiguraci
  - Spravuje proces událostí a procesy vstupů
- **Proces událostí**
  - Centrální zpracování událostí
  - Spouští kód a stavové automaty
- **Procesy vstupů**
  - Polling

ESP32 Device Architecture



Node System

- Aplikace je složena z propojených nodů
- Event-driven zpracování

Tok dat:

$$\text{Input} \rightarrow \text{Event} \rightarrow \text{Logic} \rightarrow \text{Event} \rightarrow \text{Output}$$

- Input: GPIO, Timery, MQTT
- Middle: FSM, Vlastní kód, GPIO
- Output: GPIO, MQTT

Vlastní TCP Protokol

- Jednoduchá binární implementace
- Formy zabezpečení
  - Žádné
  - Kryptohash
  - SSL/TLS

Příklad formátu paketu

(Packet Type: 4B) (Payload Size: 4B) (Payload: N)

Typy:

- Success (0x0)
- Failure (0x1)
- Config (0x2)

Spouštění kódu

Možnosti:

- Lua
- Micropython (embed verze)
- Nativní binární kod

Děkuji za pozornost