FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

Domácí automatizace – Home Assistant

Západočeská Univerzita V Plzni
Katedra Kybernetiky
Semestrální práce - HKUI

Ondřej Sluka
2023-01-17

1 Úvod

Domácí automatizace je obor zabývající se nasazením takzvaných chytrých zařízení v prostředí domácnosti za účelem dosažení stavu, kdy jsou určité běžné úkoly prováděny počítačovým systémem namísto člověka. Vedlejším účinkem provozování takového systému je obvykle i umožnění vzdáleného ovládání zařízení v domácnosti.

Tato práce přiblíží základní princip funkce běžného systému domácí automatizace na příkladu svobodného software vyvíjeného komunitou Home Assistant. Bude se zabývat především backendovou částí, protože její funkce je pro automatizaci důležitější než prezentační část, tedy frontend. Bude popsán datový model umožňující integraci chytrých zařízení různých druhů od různých výrobců do jednoho automatizačního systému a dostupné prostředky pro tvorbu automatizačních pravidel.

2 Home Assistant

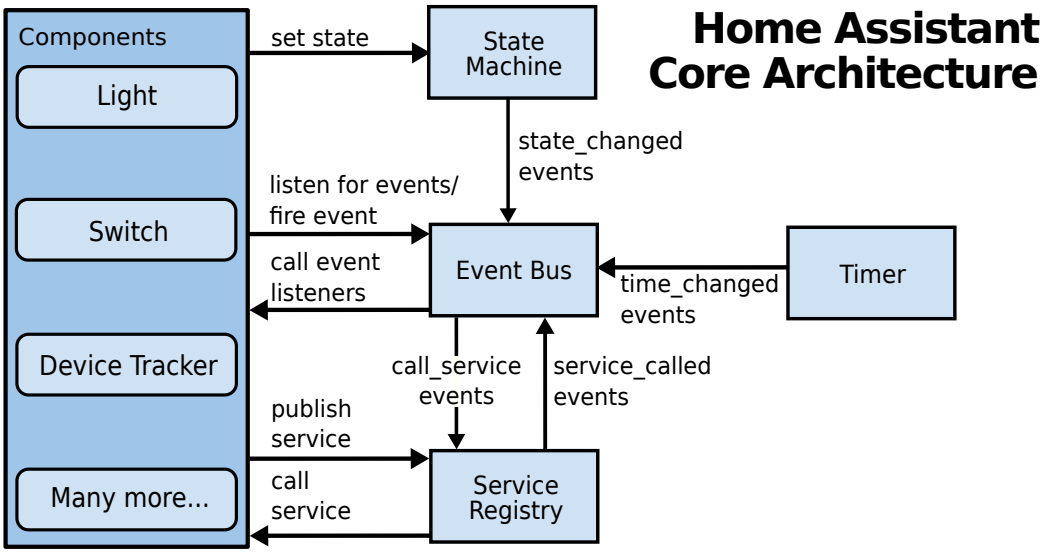
Home Assistant je svobodný software[1] distribuovaný pod licencí Apache určený k nasazení na server (zařízení NAS, jednodeskový počítač Raspberry Pi apod.) v automatizované domácnosti. Cílem projektu je poskytovat centrální řídicí systém pro chytré domácnosti fungující zcela lokálně, bez závislosti na cloudové infrastruktuře. Systém obsahuje softwarové moduly pro připojení široké škály zařízení různých výrobců – takzvané integrace.

Home Assistant má dvě základní části: frontend a backend. *Frontend* slouží k prezentaci informací o stavu chytré domácnosti, analýze historických dat, ručnímu ovládání a konfiguraci v prostředí webového prohlížeče, zatímco *backend* zajišťuje komunikaci s chytrými zařízeními, záznam dat o stavu systému do relační databáze a spouštění automatizačních pravidel.

Konfigurace backednové části se provádí především upravováním textového souboru `configuration.yaml`. Některé softwarové komponenty lze konfigurovat též interaktivně z frontendu, nebo dokonce výhradně z frontendu. Jde především o komponenty komunikující s chytrými zařízeními, pro které je často užitečná možnost zobrazit uživateli během párovacího procesu různé ovládací prvky v závislosti na zjištěných informacích o hardwaru připojovaného zařízení.

[1] `https://www.gnu.org/philosophy/free-sw.cs.html`

2.1 Architektura backendu



Obrázek 1: Architektura backendu Home Assistant [3]

Na nejnižší úrovni je backend tvořen několika funkčními bloky. Ústředním blokem je *sběrnice událostí*. Každý z bloků může vytvářet události a zaregistrovat callback funkci, která se zavolá při příchodu specifikované události.

Stavový automat ukládá informace o stavech. Jako klíč je použito `entity_id`, stavový automat jej ale nijak nevaliduje. Ke každému `entity_id` je uložen jeden stav (v případě Home Assistant jde o textový řetězec o maximální délce 255 znaků) a kolekce atributů (párů jméno–hodnota). Při změně stavu je vytvořena událost `state_changed`. Příklad takové události lze nalézt na obrázku 2 na následující straně. Teplotní čidlo `sensor.kitchen_temperature` změnilo svůj stav z „21.25“ na „21.18“. Ve frontendu se díky atributu `unit_of_measurement` a české lokalizaci zobrazuje „21,18 °C“.

K provádění akcí slouží *služby*. Jednotlivé komponenty publikují služby, které podporují a ostatní komponenty (včetně frontendu a automatizací) mohou tyto služby volat.

```yaml
event_type: state_changed
data:
  entity_id: sensor.kitchen_temperature
  old_state:
    entity_id: sensor.kitchen_temperature
    state: "21.25"
    attributes:
      state_class: measurement
      unit_of_measurement: °C
      device_class: temperature
      friendly_name: Kitchen temperature
    last_changed: "2023-01-16T16:52:19.449845+00:00"
    last_updated: "2023-01-16T16:52:19.449845+00:00"
    context:
      id: 01GPXQXYVSZT0BPXEDXEVDND7Q
      parent_id: null
      user_id: null
  new_state:
    entity_id: sensor.kitchen_temperature
    state: "21.18"
    attributes:
      state_class: measurement
      unit_of_measurement: °C
      device_class: temperature
      friendly_name: Kitchen temperature
    last_changed: "2023-01-16T16:57:19.422930+00:00"
    last_updated: "2023-01-16T16:57:19.422930+00:00"
    context:
      id: 01GPXR73SYCSKW9N3R2XC146MS
      parent_id: null
      user_id: null
origin: LOCAL
time_fired: "2023-01-16T16:57:19.422930+00:00"
context:
  id: 01GPXR73SYCSKW9N3R2XC146MS
  parent_id: null
  user_id: null
```

Obrázek 2: Příklad události typu `state_changed` (data serializována do formátu YAML)

2.1.1 Entity

Základním prvkem v automatizačním systému je entita. Jde o abstrakci nad interním stavovým automatem. Každá entita má jednoznačný identifikátor (`entity_id`) a v čase proměnný stav a atributy.

Jednotlivé entity jsou poskytovány různými integracemi a reprezentují jednotlivé datové body připojených chytrých zařízení. Existují i integrace, které poskytují entity, jejichž stav závisí na stavu jiných entit. Například integrace „Integration – Riemann sum integral“ sleduje stav zdrojové entity a aproximuje plochu pod grafem závislosti jejího stavu na čase. To je užitečné například pro určování energie spotřebované zařízením, které vystavuje pouze entitu odpovídající okamžitému příkonu.

2.1.2 Služby

Pokud k entitě přísluší nějaký ovládací prvek (například chytrý vypínač jde zapnout a vypnout), je implementován jako služba. Tato služba obvykle přijímá jako parametr `target` ID entity, na které se má akce provést. Například následující volání služby `switch.turn_on` sepne relé ovládající oběhové čerpadlo. Odpovídající entita `switch.dhw_circulator_relay` následně (v případě úspěšného provedení akce) přejde do stavu „on“.

```
service: switch.turn_on
data: {}
target:
  entity_id: switch.dhw_circulator_relay
```

2.1.3 Komunikační protokoly

Komunikace s chytrými zařízeními je zajišťována příslušnými integracemi. Home Assistant obsahuje takových integrací přes 1000 [5], přičemž lze doinstalovat i libovolné neoficiální integrace. Důsledkem je velmi široká škála používaných komunikačních protokolů. Pro uživatele (tvůrce automatizací) nejdůležitější je rozdělení do dvou kategorií podle toho, která strana zahajuje přenos dat – *polling* a *push*. V *polling* situaci se Home Assistant periodicky dotazuje chytrého zařízení na jeho stav, zatímco v situaci *push* je změna stavu okamžitě reportována samotným zařízením. Důsledkem je rychlejší odezva zařízení nabízejících push komunikaci. Je tak snazší zajistit, že automatizace provede svůj úkol včas. [6]

2.2 Automatizace

Podstatou celého systému je poskytnout základ pro jednoduchou tvorbu automatizačních pravidel. Každé automatizační pravidlo je tvořeno 3 částmi: spouští (`trigger`), podmínkami (`condition`) a akcemi (`action`).

Provádění pravidla je iniciováno *spouští*. Obecně jde o událost, díky abstrakcím ale lze v konfiguraci specifikovat různé druhy spouští (state trigger, event trigger, time pattern trigger apod. [7]).

Po spuštění pravidla následuje ověření *podmínek*. Na rozdíl od spouští, které reagují na události, podmínky ověřují stavy. K dispozici jsou například state condition, numeric state condition a time condition a logické spojky AND, OR a NOT [8]. Pokud kterákoli z podmínek není splněna, provádění pravidla se ukončí.

Jestliže jsou všechny podmínky splněny, začne provádění *akcí*. Jde o sekvenci příkazů (vytvoření události, volání služby, čekání apod.). K dispozici jsou řídicí struktury `condition` (podmínka IF), `repeat` (smyčky WHILE, FOR, FOR EACH) a `choose` (SWITCH a CASE).

Automatizační pravidla se zapisují do konfiguračního souboru `automations.yaml`. Součástí frontendu je interaktivní editor, textová reprezentace ve formátu YAML je ale přehlednější.

2.2.1 Tvorba automatizací

Díky promyšlené architektuře systému je tvorba pravidel poměrně intuitivní. Dobře vytvořená pravidla mají velmi nízké nároky na systémové prostředky a není proto problém mít několik stovek takových pravidel. Efektivní pravidlo například neověřuje stav určité entity každou sekundu, ale je spouštěno pouze při změnách stavu této entity.

Následuje okomentovaný příklad automatizačního pravidla:

```
alias: Obehove cerpadlo noc
trigger:
  # 1. spoušť - při přepnutí elektronického zabezpečovacího systému
  # do režimu noc na dobu minimálně 30 sekund
  - platform: state
    entity_id: binary_sensor.alarm_night
    from: "off"
    to: "on"
    for:
      seconds: 30
    # id spouště -- v těle automatizace lze zjistit,
    # která spoušť ji spustila
    id: night on
```

```
  # 2. spoušt - při odstřežení
  - platform: state
    entity_id: binary_sensor.alarm_night
    from: "on"
    to: "off"
    id: night off
condition:
  - or:
      # Buď došlo k zastřežení a je noc
      - and:
          - condition: trigger
            id: night on
          - condition: time
            after: "20:00"
            before: "03:00"
      # Nebo došlo k odstřežení
      - condition: trigger
        id: night off
action:
  - choose:
      # pokud došlo k zastřežení, vypni čerpadlo
      - conditions:
          - condition: trigger
            id: night on
        sequence:
          - service: switch.turn_off
            data: {}
            target:
              entity_id: switch.dhw_circulator_control
      # pokud došlo k odstřežení, zapni čerpadlo
      - conditions:
          - condition: trigger
            id: night off
        sequence:
          - service: switch.turn_on
            data: {}
            target:
              entity_id: switch.dhw_circulator_control
    default: []
```

3 Závěr

Home Assistant nabízí otevřenou platformu pro domácí automatizaci. Díky abstrakci je možné pracovat s velmi širokou škálou zařízení. Na rozdíl od komerčních řešení je veškeré zpracovávání uživatelových dat prováděno na jeho vlastním serveru bez jakékoli závislosti na cizí infrastruktuře. Výsledkem je velmi robustní systém, který může sloužit i pro záznam různých statistik – například informací o spotřebě elektrické energie.

Odkazy

1. HOME ASSISTANT. *Home Assistant logo* [online] [cit. 2023-01-16]. Dostupné z: `https://github.com/home-assistant/home-assistant.io/blob/c4c6030149d91b788adcc7fb6dff4c7d888d2365/source/images/home-assistant-logo.svg`.
2. HOME ASSISTANT. *Home Assistant documentation* [online] [cit. 2023-01-16]. Dostupné z: `https://www.home-assistant.io/docs/`.
3. HOME ASSISTANT. *Home Assistant Core Architecture* [online] [cit. 2023-01-16]. Dostupné z: `https://github.com/home-assistant/developers.home-assistant/blob/bb182b3c9a52e228deeddc3a63950ec6b1923fd8/static/img/en/architecture/ha_architecture.svg`.
4. HOME ASSISTANT. *Core Architecture* [online] [cit. 2023-01-16]. Dostupné z: `https://developers.home-assistant.io/docs/architecture/core`.
5. HOME ASSISTANT. *Home Assistant Components source code* [online] [cit. 2023-01-17]. Dostupné z: `https://github.com/home-assistant/core/tree/b7f484a84f6b136d6d0d30439376d9e29b38c346/homeassistant/components`.
6. SCHOUTSEN, Paulus. *Classifying the Internet of Things* [online]. 2016-02-12 [cit. 2023-01-17]. Dostupné z: `https://www.home-assistant.io/blog/2016/02/12/classifying-the-internet-of-things/`.
7. HOME ASSISTANT. *Automation Trigger* [online] [cit. 2023-01-16]. Dostupné z: `https://www.home-assistant.io/docs/automation/trigger/`.
8. HOME ASSISTANT. *Conditions* [online] [cit. 2023-01-16]. Dostupné z: `https://www.home-assistant.io/docs/scripts/conditions/`.