Cisco IOT systém

Konkrétně Cisco IoT System snižuje složitost digitalizace pro průmysl. Jak je znázorněno na obrázku, Cisco IoT System identifikuje šest technologických pilířů, které pomáhají zjednodušit a zabezpečit nasazení IoT:

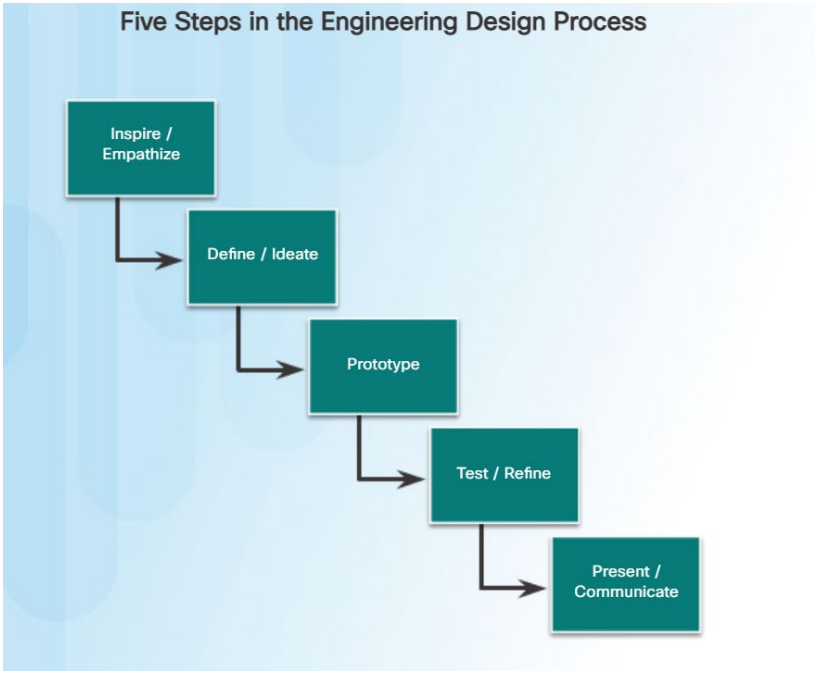
- Připojení k síti
- Fog Computing
- Kybernetická a fyzická bezpečnost
- Analýza dat
- Řízení a automatizace
- Platforma pro aktivaci aplikací

Návrh řešení

Při navrhování nových řešení a produktů je nejlepší použít osvědčenou metodu. Inženýrské týmy často používají proces inženýrského návrhu znázorněný na obrázku.

Proces inženýrského návrhu je řada kroků používaných inženýry při práci na řešení problému. Řešení se často týká návrhu produktu (zařízení nebo počítačového kódu), který splňuje specifická kritéria nebo splňuje konkrétní úkol.

Těchto pět kroků de může cyklicky opakovat tolikrát, kolikrát je potřeba, aby došlo ke zlepšení procesu návrhu.



Plánování zabezpečení

Spíše než přidáním bezpečnosti dodatečně, *musí být zabudováno do jakéhokoli zařízení IoT již od fáze návrhu.* Při navrhování zařízení IoT od začátku se ujistěte, že obsahuje funkce pro usnadnění aktualizací softwaru. Všechna ladící zadní vrátka nebo účty by měly být z kódu odstraněny.

Zabezpečení prostřednictvím předpokladu, že útočníci nikdy nenajdou skrytá zadní vrátka je falešné a nelze mu věřit. Pracujte také s myšlenkou, že útočníci najdou všechna zakódovaná zadní vrátka, nebo účty a použijí je ke kompromitaci systému.

Při práci s předem vyrobenými nebo běžně dostupnými zařízeními je třeba dbát ještě větší opatrnosti. Níže je uvedeno několik položek ke kontrole:

Výchozí hesla: Všechny výchozí účty musí být zakázány. Pokud deaktivace výchozích účtů není možná, ujistěte se, že hesla pro tyto účty jsou změněna na silná hesla.

UPnP: Universal Plug and Play (UPnP) je protokol navržený tak, aby pomohl zařízením umístěným za routerem NAT automaticky se zpřístupnit z internetu.

Vzdálená správa: Některá zařízení IoT provozují služby vzdálené správy, pomocí protokolů Telnet nebo SSH. I když ne všechny služby správy samy o sobě jsou ze své podstaty chybné, všechny umožňují sondování. Obecným pravidlem je zakázat služby vzdálené správy na zařízeních vystavených internetu. V rámci sítě LAN lze použít šifrovanou službu správy, jako je SSH, ale pouze v případě, že účet a heslo nejsou výchozí. Telnet je obzvláště problematický, protože nešifruje provoz a měl by být ve všech situacích zakázán.

Aktualizace a opravy softwaru: Zařízení IoT musí dostávat pravidelné aktualizace stejně jako stolní počítače, notebooky a mobilní zařízení. Ujistěte se, že používané zařízení IoT má vždy nejnovější software a sledujte aktualizace webu výrobce.

Šifrovaná komunikace a certifikáty: Mnoho běžných IoT zařízení nepodporuje pokročilé funkce zabezpečení, jako je šifrování nebo používání certifikátů. Pokud zařízení IoT takové pokročilé bezpečnostní funkce podporuje, měly by být funkce povoleny a používány.

Fyzická bezpečnost: Je také důležité vzít v úvahu fyzické zabezpečení zařízení. I když lze pro fyzické zabezpečení hotových IoT zařízení udělat jen málo, důrazně se doporučuje zabezpečit umístění fyzického nasazení, kdykoli je to možné.

Proces vytváření IoT systému

Project Overview

Mnoho produktů IoT se rodí z nutnosti; problém musí být vyřešen a řešení lze dosáhnout pomocí zařízení IoT. **Prvním krokem při vytváření produktu IoT je identifikace problému**, který lze vyřešit pomocí zařízení IoT.

Příklad návrhu

Zvažte tento scénář: chcete v Kalendáři Google zobrazit časy západu a východu slunce u vás doma. Dalo by se to vyhledat na specializovaném webu, ale není v tom žádná legrace. Rozhodnete se postavit zařízení, které tuto práci zvládne.

Problém: Sestavte zařízení, které dokáže snímat množství světla venku a rozhodnout, zda nastal západ nebo východ slunce. Toto zařízení provede specifickou akci, když je detekován východ nebo západ slunce.

Řešení: Zařízení musí obsahovat světelný senzor pro měření množství světla venku. Zařízení bude také napájeno baterií, aby bylo flexibilnější. Protože zařízení musí při detekci východu nebo západu slunce provést akci, použijeme internetovou službu **IFTTT** (If-this-then-that), která se snadno propojí s kalendářem Google. Zařízení proto musí umět komunikovat s internetem.

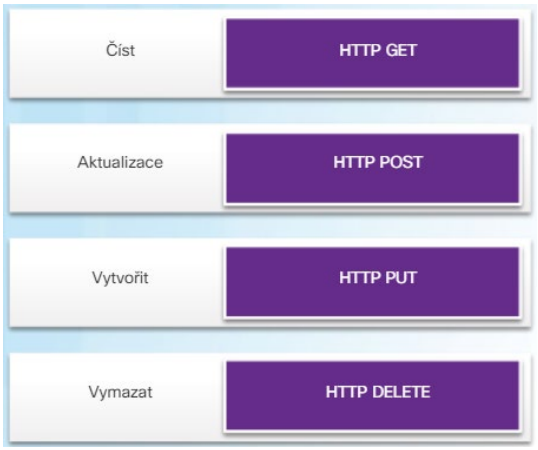
Zařízení musí být také schopno se rozhodnout na základě události (východ nebo západ slunce). Protože je vyžadováno rozhodnutí, bude potřeba napsat nějaký kód. Jako mozek zařízení a ke spuštění kódu použijeme Raspberry Pi. Arduino je také nezbytné, protože má analogové piny, které nejsou v Raspberry Pi. K Arduinu se připojí fotorezistor a použije se k detekci změn v množství světla.

Fotorezistor funguje tak, že snižuje svůj vnitřní odpor mezi vysokými a nízkými ohmy v závislosti na množství světla, které na něj dopadá. Pokud není detekováno žádné světlo, je odpor vysoký a fotorezistorem protéká velmi malý proud. V tomto konkrétním případě více světla snižuje odpor, což zase zvyšuje proud protékající obvodem a výsledný pokles napětí lze měřit pomocí Arduina. Kód běžící v Raspberry Pi bude sledovat úroveň napětí na analogovém vstupním pinu Arduina, který je připojen k fotorezistoru, a na základě úrovně napětí rozhodne, zda došlo k východu nebo západu slunce. Kód poté odešle zprávu IFTTT, která spustí akci k zaznamenání události (východu nebo západu slunce) a času, kdy k ní došlo.

REST API v systému IOT

Často komunikují přes Hypertext Transfer Protocol (HTTP) pomocí stejných metod HTTP (**GET**, **POST**, **PUT,** a **DELETE**). Metody HTTP také používají webové prohlížeče k načítání webových stránek nebo k odesílání dat na webové servery.

Systémy RESTful používají **Uniform Resource Identifiers** (**URI**) k reprezentaci svých služeb pro externí systémy. Systém může například poskytnout URI pro dotazování na uživatelské profily reprezentované /people/NAME. Takovou službu si mohou vyžádat externí systémy pomocí standardních HTTP metod, jako je GET /people/michael pro **příjem datové sady** Michaelova uživatelského profilu nebo POST /people/michael pro **aktualizaci** Michaelova profilu novými daty.



https://maker.ifttt.com/trigger/SunRise/with/key/, kde:

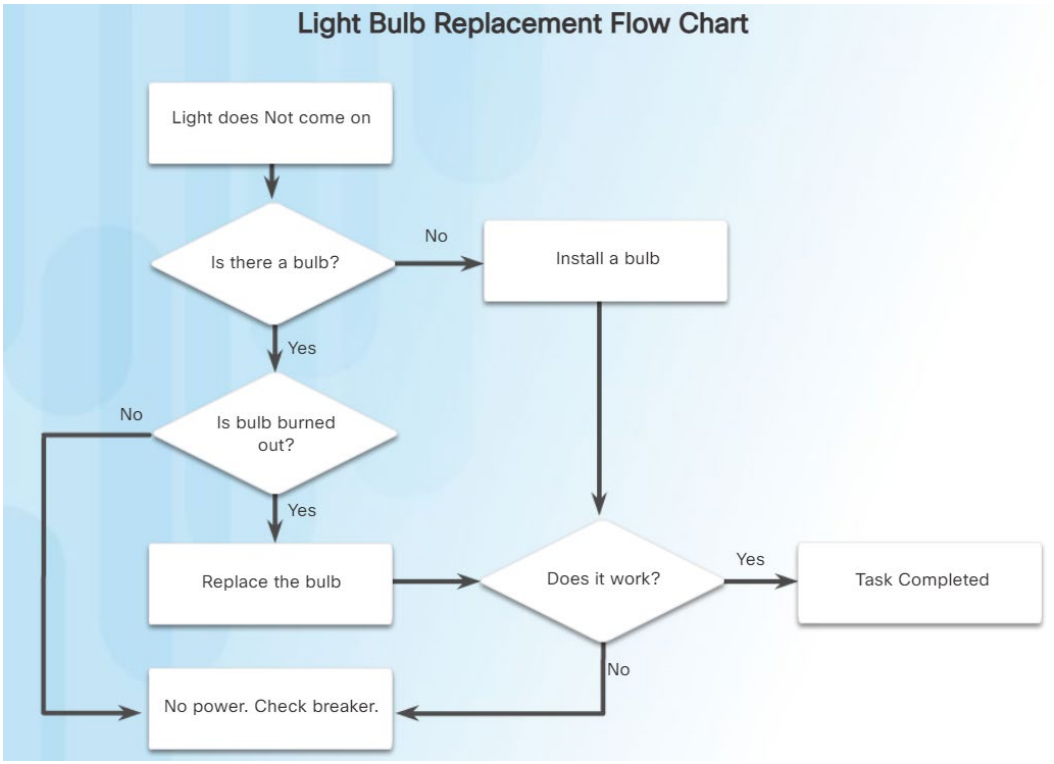
https://	používaný protokol, v tomto případě je to HTTPS
maker.ifttt.com	URL adresa serveru. Je nezbytná k identifikaci IFTTT serveru v rámci Internetu
trigger/SunRise/with/key	speciální adresa IFTTT URI vytvořené a namapovaná pro zpracování událostí východu slunce
KEY VALUE	tajný klíč poskytovaný IFTTT, aby bylo zajištěno, že speciální URI mohou používat pouze autorizovaná zařízení a aplikace. Tento klíč se bude u různých účtů lišit

```
import requests # web http requests library

# copy here your personal Maker Secret Key from https://ifttt.com/maker
#  e.g.: iFTTTMakerSecretKey = "lBX_vJGs2xa******************P7bakzuQq"
iFTTTMakerSecretKey = "lBX_vJGs2xa******************P7bakzuQq"

# The IFTTT Maker Channel URLs as configured in your IFTTT recipes for SunRise and
SunSet
iFTTTSunRiseURL = "https://maker.ifttt.com/trigger/SunRise/with/key/"
+ iFTTTMakerSecretKey
iFTTTSunSetURL  = "https://maker.ifttt.com/trigger/SunSet/with/key/"
+ iFTTTMakerSecretKey

r = requests.get(iFTTTSunRiseURL)
# if the status_code is different from 200, something went wrong:
print ("The resulting HTTP GET status code was " + str(r.status_code))
```

Vývojové diagramy

Vývojové diagramy jsou používané k reprezentaci procesů nebo pracovních postupů. S využitím různých tvarů, rámečků a spojovacích šipek představuje vývojový diagram postup řešení daného problému. Vývojové diagramy se běžně používají k reprezentaci programů, algoritmů nebo jakéhokoli uspořádaného procesu v různých oblastech. Existují různé typy vývojových diagramů. Každý typ vývojového diagramu má svou vlastní sadu rámečků a konvencí. Dvě nejběžnější políčka ve vývojovém diagramu jsou následující:

- **Proces** (obvykle nazývaný **aktivita**), který je reprezentován **obdélníkem**
- **Rozhodnutí**, které je obvykle reprezentováno tvarem **diamant**



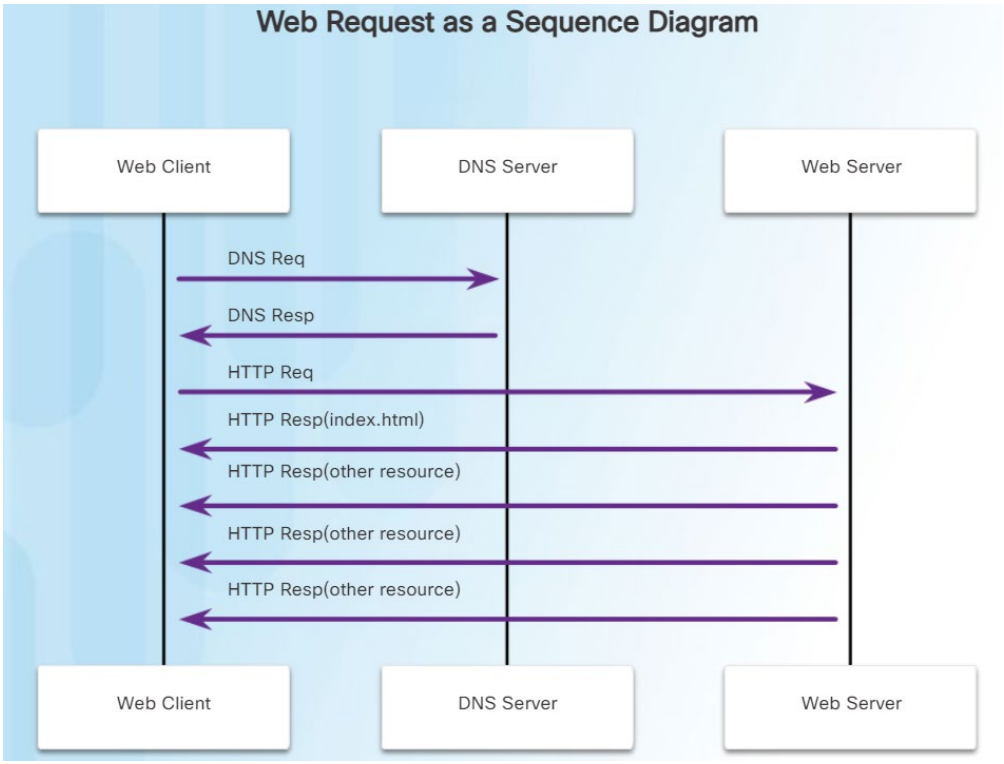
Elektronická schémata

Elektronické schéma (také známé jako schéma zapojení, elektrické schéma) je grafické znázornění elektronického obvodu. Schematický diagram znázorňuje součásti a jejich zapojení pomocí standardizovaných symbolů. Elektronická schémata jsou široce používána v IoT pro návrh a konstrukci obvodů (jako je rozložení desky s plošnými spoji) a údržbu elektrických a elektronických zařízení.

Symboly používané k reprezentaci součástí se v průběhu let měnily, ale nyní jsou mezinárodně standardizovány.

Sekvenční diagramy

Sekvenční diagram, nazývaný také diagram událostí nebo scénář událostí, se používá k reprezentaci interakcí mezi entitami podél časové osy. Sekvenční diagramy jsou účinným způsobem, jak znázornit posloupnost zpráv vyměňovanými mezi entitami. Jsou běžnou volbou při reprezentaci různých typů komunikace mezi zařízeními.



Kódování

Program běžící na Raspberry Pi je to, co dává zařízení inteligenci. Je to kód, který umožní zařízení analyzovat data přijatá na analogovém pinu Arduino A0 a rozhodnout, zda se jedná o západ nebo východ slunce. Protože Raspberry Pi nemá žádné analogové piny, používá se také Arduino. Analogové vstupní piny obsažené v Arduinu jsou požadavkem pro měření skutečné úrovně poklesu napětí v obvodu děliče napětí způsobeného změnou odporu ve fotorezistoru. Chcete-li mít data přenosu Arduino z jeho pinů na Raspberry Pi přes USB, musí být v Arduinu nainstalován speciální firmware.

Běžnou volbou firmwaru Arduino pro přenos informací z Arduina přes jeho USB port je **Firmata**. Firmata je obecný protokol pro komunikaci s mikrokontroléry ze softwaru na hostitelském počítači. Je určen pro práci s jakýmkoli softwarovým balíčkem počítače. V tomto příkladu je Firmata použita, aby umožnila Arduinu komunikovat s Raspberry Pi a vyměňovat si informace přijaté na jeho GPIO pinech.

```
# import knihovny firmata pro čtení hodnoty přímo z Arduina
from pyfirmata import Arduino, util

analog_value = None

# inicializace sériové komunikace na správném rozhraní
board = Arduino('/dev/ttyACM0')
util.Iterator(board).start()

# nastavení čtení z analogového pinu potenciometru
board.analog[0].enable_reporting()

...

# čtení z nanalogového pinu A0 do proměnné
analog_value = board.analog[0].read()
```

Prototyp a testování

Pro prototypování lze použít mnoho platforem. **Nepájivé pole** je běžnou možností, protože umožňuje vytvoření fyzického prototypu bez nutnosti pájení. Packet Tracer (PT), výkonný síťový simulátor s podporou IoT, je také skvělým způsobem prototypování, protože podporuje akční členy, senzory a kontrolery. Packet Tracer také obsahuje řadu plně konfigurovatelných síťových zařízení a umožňuje také prototypování síťové komunikace.

V rámci procesu prototypování je třeba rozhodnout o fungování systému IoT. Elektronické schéma, sekvenční diagram a vývojový diagram jsou dobré nástroje pro reprezentaci, formalizaci návrhu a provoz prototypu. Elektronické schéma poskytuje reprezentaci elektronického obvodu. Sekvenční diagram se používá k reprezentaci interakcí mezi entitami podél časové osy. Vývojový diagram se používá k reprezentaci procesů nebo pracovních postupů.

Formalizace a dokumentace

Když je prototyp připraven, je důležité zdokumentovat veškeré změny v detailech konstrukce. Prohlédněte si dokumentaci (elektrická schémata, sekvenční diagramy a vývojové diagramy), abyste se ujistili, že je v souladu s hotovým prototypem.

Obrázky hotového zařízení jsou také žádoucí, protože pomáhají při vyobrazení zařízení. Doporučuje se také krátký text popisující činnost zařízení. Dokumentace je důležitá nejen pro budoucí použití, ale také pro situace, kdy mají být vytvořeny marketingové materiály nebo patenty a ochranné známky.

Stručně řečeno, dokumentace by měla obsahovat:

- Definice problému
- Přehled řešení
- Typ senzorů a aktuátorů, které se používají
- Typ použitého kódu

- Typ síťových připojení používaných ke komunikaci
- Popis používaných cloudových služeb a způsobu jejich použití
- Zda se jedná o udržitelný obchodní model
- Logický diagram od konce ke konci
- Sekvenční diagram
- Vývojový diagram
- Schéma elektrického obvodu