ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Manager síťových připojení jako modul pro robota NAO
Student:	Tomáš Jelínek
Vedoucí:	Ing. Miroslav Skrbek, Ph.D.
Studijní program:	Informatika
Studijní obor:	Informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Navrhněte a implementujte modul pro robota NAO, který bude umožňovat management síťových připojení. Modul musí obnovit zvolené připojení při jeho náhlém výpadku, volit různé WiFi sítě a hlasově informovat o stavu sítí. Konfigurace sítí bude uložena v konfiguračních souborech. Současně s tím řešte i otázku počítačové bezpečnosti, jako je nastavení firewallu, povolení přístupu pro určité IP adresy, porty, sítě. Předpokládá se podpora jak češtiny, tak angličtiny pro hlasový výstup. Rozsah implementace konzultujte s vedoucím práce.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 29. listopadu 2016

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra počítačových systémů

Bakalářská práce

Manager síťových připojení jako modul pro robota NAO

Tomáš Jelínek

Vedoucí práce: Ing. Miroslav Skrbek, Ph.D.

26. června 2017

Poděkování

Děkuji vedoucímu práce Ing. Miroslavu Skrbkovi, Ph.D. za cené rady a odborné vedení mé práce. Poděkování patří také mým rodičum za poskytnutí zázemí a podpory při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 26. června 2017 .....

Odkaz na tuto práci

Jelínek, Tomáš. *Manager síťových připojení jako modul pro robota NAO*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Práce se zabývá návrhem a implementací modulu pro robota NAO, který umožňuje management síťových připojení. Navržený modul je schopen automatické obnovy zvoleného WiFi připojení při jeho náhlém výpadku, volit různé WiFi sítě a hlasově informovat o stavu sítí. Hlasový výstup podporuje jak český, tak anglický jazyk. Práce také řeší otázky počítačové bezpečnosti, zejména zabezpečení síťového provozu přes Wifi pomocí firewallu. Funkčnost výsledného modulu implementovaného v jazyce c++ byla otestována přímo na robotovi NAO.

Klíčová slova robot NAO, NAO modul implementace, framework NAOqi, připojení WiFi, linux firewall, wpa_supplicant, c++

Abstract

The thesis deals with design and implementation of modules for robot NAO, which allows management of network connections. The designed module is capable of automatic recovery of the selected WiFi connection at his sudden loss, choose different WiFi networks and use voice to inform about state of the network. Voice output supports both Czech and English. The thesis also solves computer security issues, especially the security of network traffic via Wifi using the firewall. The functionality of the resulting module implemented in c++ was tested directly on the NAO robot.

Keywords robot NAO, NAO modul implementation, framework NAOqi, WiFi connection, linux firewall, wpa_supplicant, c++

Obsah

Seznam obrázků

Úvod

Robotika se bezpochyby stává oborem blízké budoucnosti. S autonomními humanoidními roboty se už můžeme setkat nejenom v armádě, vědeckých laboratořích a na univerzitách, ale i v běžném životě. Živým důkazem nechť je robot NAO (obrázek 0.1), vyvíjený firmou SoftBank Robotics, který bezchybně zvládá obsluhovat zákazníky v jedné z bank v Tokiu.

V současnosti je robot NAO nejčastěji využíván v akademickém prostředí. Není proto žádným překvapením, že tohoto robota vlastní i Fakulta informačních technologií ČVUT. NAO zde slouží k propagaci fakulty při dnech otevřených dveří a dalších událostech pořádaných fakultou. Studenti se mohou s robotem seznámit v rámci semestrálních prací v bakalářském předmětu BI-ZIVS nebo magisterském předmětu MI-IVS. Jedním z klíčových předpokladů správného a neomezeného fungování robota je jeho připojení k WiFi síti. Který uživatel by proto nevyužil v této práci vytvořený modul pro snazší a pohodlnější připojování robota k WiFi?

Téma Manager síťových připojení jako modul pro robota NAO jsem si zvolil, neboť současné možnosti připojení robota k WiFi nejsou dostatečné. Jako hlavní nedostatek se jeví složitý proces připojování a volby WiFi sítě. Nemohu také opomenout slabiny vestavěného firewallu, který je pro potřeby práce s robotem příliš jednoduchý. Tato práce by měla být přínosem pro všechny uživatele robota NAO na naší fakultě. Při tvorbě modulu mám tak jedinečnou příležitost si osahat technologie, ke kterým bych se za normálních okolností jen těžko dostával.

Práce se zaměřuje na návrh a implementaci modulu pro připojení robota NAO k WiFi síti a analýzu možností konfigurace a zabezpečení jednotlivých sítí firewallem.

Obrázek 0.1: Robot NAO (převzato z dokumentace [1])

V první kapitole se čtenář seznámí s hlavními cíli této práce. Teoretickou část práce reprezentuje kapitola dvě s názvem Analýza, kde jsou prozkoumány možnosti a dostupné technologie potřebné pro realizaci modulu a provedena rešerše existujících řešení. Třetí kapitola obsahuje analýzu požadavků na funkčnost modulu, nejčastější případy užití a samotný návrh modulu. Ve čtvrté kapitole je popsána implementace celého modulu. Pátá kapitola se zabývá testováním modulu přímo v robotovi NAO. Práce je zakončena kapitolou Závěr shrnující výsledky práce.

Kapitola 1

Cíl práce

Cílem práce je návrh a implementace modulu pro robota NAO, který bude umožňovat management síťových připojení. Modul musí obnovit zvolené připojení při jeho náhlém výpadku, volit různé WiFi sítě a hlasově informovat o stavu sítí. Konfigurace sítí bude uložena v konfiguračních souborech. Současně s tím bude řešena i otázka počítačové bezpečnosti, jako je nastavení firewallu, povolení přístupu pro určité IP adresy, porty, sítě. Předpokládá se podpora jak češtiny, tak angličtiny pro hlasový výstup.

Kapitola 2

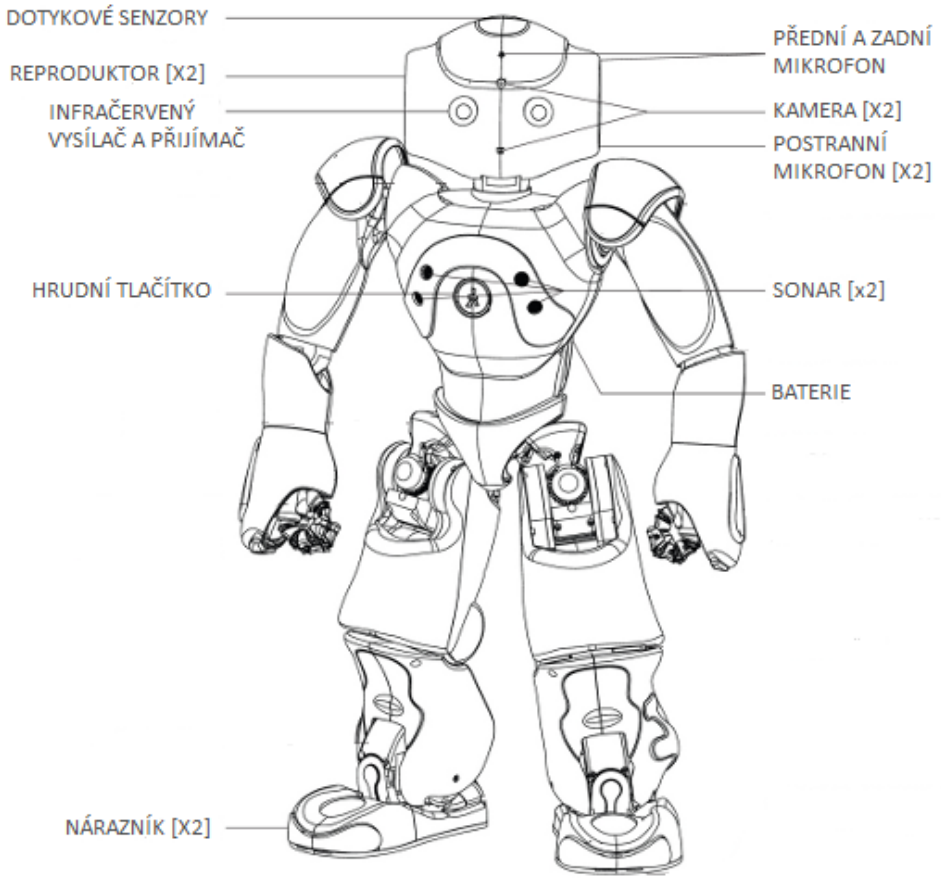
Analýza

2.1 Robot NAO

Dle technické specifikace dostupné z dokumentace [1] je robot NAO verze 4 opravdu multifunkční stroj, vybavený celou řadou technologických vymožeností. Rozmístění jednotlivých komponent na těle robota (obrázek 2.1) je navrženo tak, aby optimálně využívalo veškeré volné prostory robotova těla.

Výška 57 cm a váha 5,4 kg umožňuje snadnou manipulaci při přenášení. Základní deska umístěná v hlavě robota je osazena jednojádrovým procesorem Intel Atom Z530 o frekvenci 1,6 GHz, 1 GB RAM a 2 GB flash pamětí. Místo pevného disku využívá NAO 8 GB microSDHC paměťovou kartu. Pro připojení k síti zde slouží síťová karta s podporou WiFi standardů IEEE 802.11 b/g/n a zabezpečení WEP, WPA a WPA2. Napájení zajišťuje lithium-iontová baterie o kapacitě 2,25 Ah, která při plném nabití vydrží až 90 minut bez nutnosti připojit robota kabelem k elektrické síti. Zvukový výstup zajišťují dva stereo reproduktory umístěné v uších robota. Dále je NAO vybaven celkem čtyřmi mikrofony zabudovanými v uších, čele a týle robota, díky čemuž je schopen snadno zaznamenat zvuky přicházející ze všech směrů. Obrazové vnímání umožňují 2 kamery s maximálním rozlišením 1280x960 pixelů a frekvencí 30 snímků za sekundu. První, která se nachází na čele, snímá okolí robota ve výšce jeho očí a druhá, umístěná v jeho ústech, slouží k snímání povrchu pod jeho nohama. V trupu je pak uložen inerciální navigační systém s vlastní procesorovou jednotkou skládající se ze 2 gyroskopů a akcelerometru. Na každé straně hrudníku má NAO sonar, který pomocí ultrazvukových vln dokáže určit vzdálenost objektů od robota. Robot může využívat propracovaný systém kloubních senzorů a motorů, jež mu umožňují 25 stupňů volnosti pohybu. K interakci s okolím lze využít celkem 9 kapacitních dotykových senzorů umístěných po trojicích na temeni hlavy a zápěstích obou paží. Krom dotykových senzorů má robot ještě mechanické tlačítko na hrudi a 4 spínatelné nárazníky na špičkách chodidel. Oči a jednotlivé senzory jsou podsvíceny

barevnými LED diodami, které změnou své barvy vyjadřují aktuální stav robota. Díky v očích umístěnému infračervenému portu je NAO schopen přenosu dat prostřednictvím infračerveného záření. Pro připojení externích zařízení je možené využít USB a ethernetový port v týlu hlavy robota.



Obrázek 2.1: Schéma robota NAO (přeloženo z dokumentace [1])

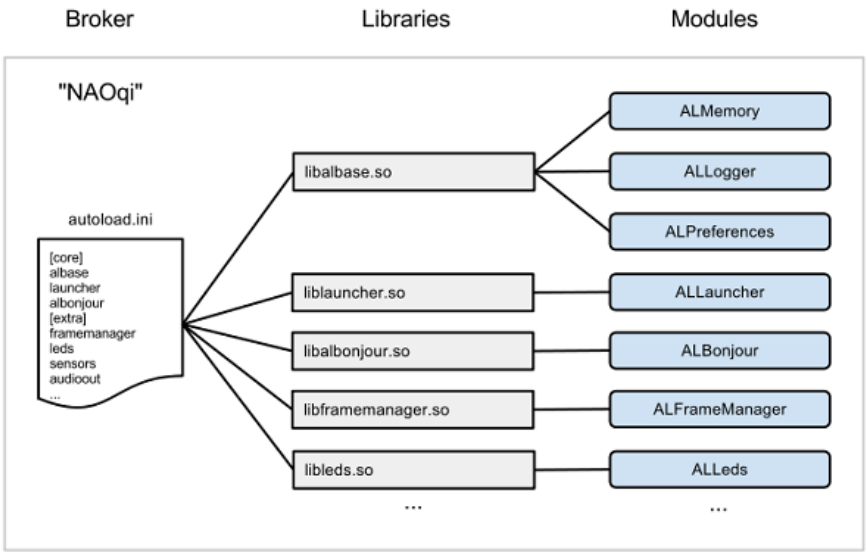
Operační systém robota NAO se nazývá NAOqi OS. Jedná se o distribuci GNU/Linux založenou na Gentoo a speciálně upravenou pro potřeby robota. Tento operační systém běží pouze v textovém režimu, neboť robot nemá možnost grafického výstupu. Z hlediska administrace lze k tomuto systému přistupovat jako k jakémukoli linuxovému serveru prostřednictvím protokolu ssh. Hlavním rozdílem oproti běžnému Gentoo serveru je přítomnost softwaru NAOqi, který běží na robotovi a ovládá jej. A právě popis frameworku NAOqi je obsahem následující sekce.

2.2 Framework NAOqi

Framework NAOqi [1] verze 2.1 je software zodpovědný za správu a ovládání robota NAO. Díky svému modulárnímu konceptu, se z něj navíc stává mocný nástroj k dalšímu programování. Samozřejmostí je zde podpora paralelního zpracování, synchronizace a interakce s událostmi. Framework podporuje vývoj aplikací pro operační systémy Windows, Linux a MacOS. Programovat je možno v jazycích Python, C++, Java a Javascript. Použití každého z těchto jazyků má svoje výhody a nevýhody, které určují, kdy který jazyk použít. Jazyk Python se obvykle používá pro programování chování robota. K tvorbě nových modulů slouží především jazyk C++ a v omezené míře i jazyk Java. Javascript je používán pro tvorbu webových aplikací využívaných robotem NAO.

Proces NAOqi

Proces NAOqi (obrázek 2.2) při svém startu načte inicializační soubor autoload.ini, který obsahuje názvy vybraných knihoven. Každá knihovna obsahuje jeden nebo více modulů robota NAO. Metody jednotlivých modulů jsou dostupné skrze objekt nazývaný Broker. Broker umožňuje nalézt moduly a jejich metody lokálně nebo po síti. Pro zavolání metody modulu je třeba vytvořit objekt Proxy, který se chová jako modul, jenž reprezentuje. Pokud je tedy například vytvořen Proxy na modul ALConnectionManager, výsledný objekt poskytuje všechny registrované metody tohoto modulu.



Obrázek 2.2: NAOqi proces (převzato z dokumentace [1])

Moduly

Každý modul je třída odvozená od třídy ALModule z frameworku NAOqi. Moduly rozdělujeme na lokální a vzdálené, podle toho, jakým způsobem jsou inicializovány. Vzdálené moduly jsou obvykle kompilované jako binární spustitelné soubory a mohou být spouštěny mimo operační systém robota. Vzdálený modul využívá Brokeru ke komunikaci s jinými moduly pomocí sítě. Tato komunikace je realizována za využití protokolu SOAP pro výměnu zpráv založených na jazyku XML. Výhodou vzdálených modulů je snadnější používání a ladění. Kvůli komunikaci po síti jsou však pomalejší a kladou si větší nároky na paměť. Lokální moduly jsou dva a více modulů spuštěné ve stejném procesu. Načtením knihoven z inicializačního souboru autoload.ini procesem NAOqi jsou vytvořeny instance těchto tříd. Ke komunikaci mezi moduly je pak využíván pouze jeden Broker. Vzhledem k tomu, že lokální moduly jsou ve stejném procesu, mohou sdílet proměnné a navzájem volat metody bez nutnosti serializace a komunikace po sítí. To umožňuje co možná nejrychlejší komunikaci a využití méně paměti. Lokální moduly je možné používat pouze uvnitř robota.

Metody a události

Naoqi umožňuje dva způsoby volání metod, synchronně a asynchronně. Synchronně volaná metoda je vykonána ve vlákně, ze kterého byla volána a umožňuje provedení další instrukce až po svém dokončení, vlákno je tedy do té doby blokováno. Všechna volání mohou skončit výjimkou, která je odchytitelná pomocí try-catch bloku. Asynchronně volaná metoda je vykonávána v novém paralelním vlákně. Díky tomu je možné vykonávat více činností naráz. Robot se tak může například pohybovat a u toho zároveň mluvit. Při asynchronním volání je vytvořeno id, pomocí kterého lze zjistit zda metoda je ještě vykonávána či nikoli.

Framework NAOqi podporuje také programování řízené událostmi. Pro reakci na událost je nutné registrovat metodu, která bude při zpětném volání vykonána jako reakce na danou událost. Příkladem nechť je nově vytvořený modul MyModule obsahující metodu onChestButtonPressed. Potom lze modulu MyModule registrovat metodu ChestButtonPressed z modulu ALChestButton se zpětným voláním onChestButtonPressed. To způsobí, že při každém stisku hrudního tlačítka, bude voláno zpětné volání onChestButtonPressed.

2.3 Metody zabezpečení WiFi sítí

V následující části práce jsou představeny nejběžněji užívané metody zabezpečení WiFi sítí. Bližší seznámení přináší představu o způsobu připojování k těmto sítím a údajích potřebných pro autentizaci a autorizaci.

WEP

Zabezpečení WEP [2] umožňuje šifrovanou komunikaci mezi dvěma zařízeními v síti a znesnadňuje tak odposlech zprávy během přenosu. K šifrování je využívána proudová šifra RC4. Klíč WEP je hexadecimální řetězec jehož délka se liší podle verze WEP zabezpečení. WEP 64-bit používá 40-bitový klíč (10 hexadecimálních číslic), před který je připojen 24-bitový inicializační vektor a dohromady tak tvoří 64-bitový RC4 klíč. Obdobně je tomu pro WEP 128-bit (26 hexadecimálních číslic) a WEP 256-bit (58 hexadecimálních číslic). Zabezpečení WEP již není bezpečné, neboť existuje postup pasivního útoku, pomocí kterého lze odposloucháváním sítě získat RC4 klíč. Tuto metodu se tedy doporučuje používat pouze v případě, že přípojné zařízení neumožňuje metodu jinou.

WPA-PSK

Zabezpečení WPA-PSK [3] bylo navrženo jako náhrada již překonaného zabezpečení WEP. Přenášená data jsou šifrována pomocí stejné proudové šifry RC4, jakou používal WEP. WPA-PSK přidává autentizaci uživatelů pomocí předsdíleného klíče. Pro připojení k takto zabezpečené síti musí uživatel zadat 8 až 63 znaků dlouhé heslo. Na základě hesla je vygenerován 256-bitový klíč používaný pro šifrování a dešifrování. Dodatečné zabezpečení přináší protokol TKIP, který dynamicky generuje 128-bitový šifrovací klíč pro každý paket, čímž odstraňuje nedostatky inicializačního vektoru u zabezpečení WEP. I přes toto vylepšení je už dnes možné toto zabezpečení prolomit.

WPA2 personal (PSK)

Zabezpečení WPA2 personal [4] používá stejný způsob autentizace jako zabezpečení WPA-PSK. Zvýšení bezpečnosti spočívá v nahrazení protokolu TKIP protokolem CCMP. Tento protokol využívá blokovou šifru AES, která šifruje 128-bitový datový blok 128-bitovým šifrovacím klíčem. Každý paket je zašifrován pomocí jedinečného klíče, tudíž nedochází k opětovnému používání stejných klíčů. Protokol CCMP zajišťuje, že pouze oprávněné osoby mohou přistupovat k informacím a poskytuje důkaz o pravosti uživatele.

WPA2 enterprise

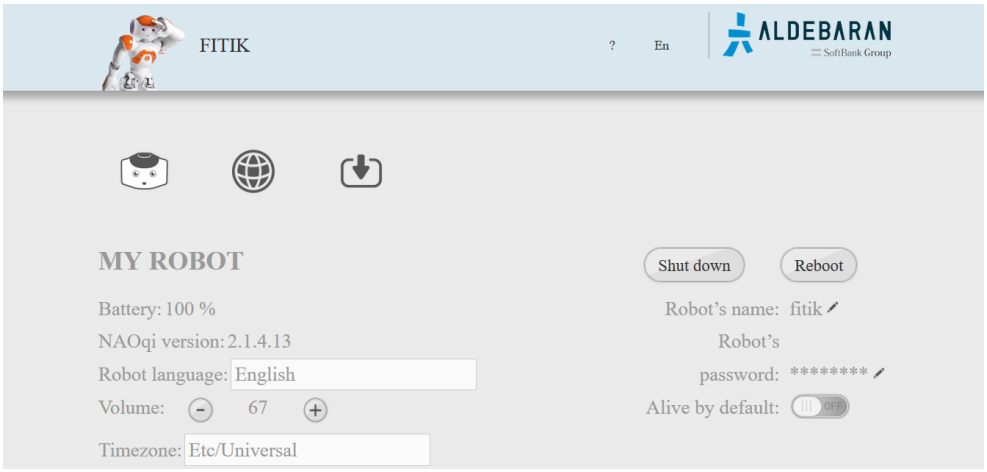
Zabezpečení WPA2 enterprise [5] používá k ověření identity uživatele RADIUS server. Zabezpečení využívající předsdílený klíč (PSK) používají pro autentizaci uživatelů jedno jedinečné globální heslo. Při autentizaci pomocí RADIUS serveru má každý uživatel heslo vlastní. Autentizaci zde zajišťuje protokol EAP. Nejběžnější konfigurací protokolu EAP je metoda PEAP s MSCHAPv2, se kterou je pro autentizaci po uživateli požadováno jeho uživatelské jméno a heslo, pro ověření proti RADIUS serveru. Druhou možností je metoda EAP-TLS, která k ověření identity využívá osobní certifikát uživatele. Šifrování je stejně jako u zabezpečení WPA2 personal zajištěno protokolem CCMP.

2.4 Současné možnosti připojování k WiFi

Připojení k WiFi síti je klíčovým požadavkem pro bezpečné a pohodlné provozování robota. Je-li robot připojen k síti ethernetovým kabelem zezadu hlavy, pak hrozí, že se při nečekaném nebo neopatrném pohybu kabel vytrhne a dojde k poškození ethernetového portu. Pro bezpečný pohyb robota je tedy WiFi nezbytnou nutností. Připojení kabelem navíc značně znesnadňuje práci více uživatelů s robotem současně, neboť vyžaduje neustálé přepojování kabelu a přesouvání po pracovišti. Následující část práce proto popisuje současné možnosti připojování robota NAO k WiFi.

2.4.1 Webová stránka NAO

Připojování k WiFi síti zajistili vývojáři pomocí webové stránky robota NAO [1]. Nejprve je nutné připojit robota pomocí ethernetového kabelu k počítači či jinému zařízení, které toto spojení umožní. Po zapnutí robota je třeba zjistit jeho IP adresu stiskem tlačítka na hrudi. Tuto adresu lze následně zadat do webového prohlížeče. Webový server běžící na robotovi poskytuje konfigurační webovou stránku (obrázek 2.3), která je dostupná na zadané adrese. Po přihlášení je možné začít nastavovat síťová připojení. Díky využití webové stránky probíhá volba WiFi sítě v grafickém prostředí. Uživatel si může vybrat preferovanou síť ze seznamu aktuálně dostupných sítí. Pokud je síť vybrána poprvé, je ještě nezbytné nastavit odpovídající metodu zabezpečení a autentizační klíč či heslo. Na závěr už stačí jen kliknout na tlačítko Connect a připojení k WiFi je navázáno. V tomto okamžiku je možné odpojit ethernetový kabel a celý proces je dokončen.



Obrázek 2.3: Webová stránka NAO

Proces připojování k WiFi pomocí webové stránky je zdlouhavý a při výpadku nebo změně sítě je potřeba jej kompletně zopakovat. Dalším zásadním nedostatkem tohoto řešení je nutnost přihlášení se jako uživatel NAO. Ne každý uživatel robota má přístup k tomuto uživatelskému účtu, který je nezbytný pro výběr a připojení k WiFi. Webová stránka také podporuje pouze sítě se zabezpečením WEP, WPA-PSK a WPA2 personal. Nelze se tak připojit například k síti Eduroam, která se nachází v prostorách univerzity, neboť zabezpečení WPA2 enterprise není webovou stránkou podporováno.

2.4.2 Modul libwifi.so

Kromě webové stránky existuje také lokální modul libwifi.so napsaný v jazyce Python přímo pro účely výuky na Fakultě informačních technologií ČVUT. Tento modul umožňuje po zapnutí robota automatické připojení k WiFi síti FIT-1048. Tato síť je pevně zvolena a není tak možné se pomocí modulu připojit k jiné síti. Při výpadku spojení je modul schopen automaticky tento problém detekovat, a pokud je to možné, tak i připojení obnovit. O změně stavu připojení a výsledku připojování informuje robot hlasovým výstupem v českém jazyce. Trojklikem na hrudní tlačítko je možné vyvolat opětovné připojování k síti.

2.4.3 wpa_supplicant

Pro ruční připojování lze také využít službu wpa_supplicant [6], která běží jako proces na pozadí systému a řídí bezdrátová připojení pro internetové rozhraní na kterém byla spuštěna. Pomocí nástroje wpa_cli, který poskytuje

textové uživatelské rozhraní této služby lze vytvářet a konfigurovat jednotlivé sítě. Supplicant umožňuje připojení ke všem sítím dle standardu IEEE 802.11i. Po připojení k WiFi síti pomocí wpa_supplicantu je třeba ještě rozhraní přidělit IP adresu např. pomocí nástroje dhclient.

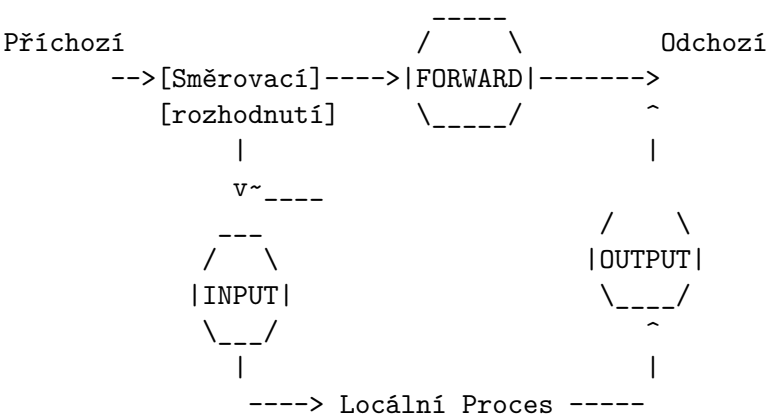
2.5 Linux firewall

Operační systém robota NAOqi OS má implementovaný vlastní firewall. Tento firewall je však velice jednoduchý a v reálném provozu těžko použitelný. Firewall může běžet ve dvou režimech. První režim s názvem development povoluje všechny porty na všech rozhraních, tudíž síťový provoz není nijak omezen. Naopak v režimu production jsou zablokovány všechny porty, kromě portu 22 pro připojení pomocí ssh na ethernetovém rozhraní.

2.5.1 Netfilter

Linuxové operační systémy s jádrem 2.4.x a novějším mají zabudovaný framework netfilter [7] umožňující filtrování síťových paketů, překlad adres a modifikování hlaviček paketů.

Pomocí nástroje iptables lze přidávat a odebírat pravidla z tabulky filtrování paketů umístěné v jádře. Tabulka obsahuje tři základní řetězce pravidel INPUT, OUTPUT a FORWARD. Řetězec INPUT zpracovává příchozí pakety, řetězec OUTPUT odchozí pakety a řetězec FORWARD, přeposílá pakety určené pro jiná zařízení.



Řetězce obsahují kaskádní seznam pravidel, která jsou vyhodnocována v pořadí jejich zápisu v řetězci. Jestliže hlavička paketu odpovídá pravidlu, potom je s paketem provedena akce uvedená v tomto pravidle a paket již není dál testován pravidly následujícími v řetězci. Existují tři různé akce. Akce ACCEPT znamená povolení průchodu paketu, akce DROP zahodí paket a

nedojde tak k jeho doručení. Poslední akcí je akce REJECT, která stejně jako akce DROP paket zahodí, ale odešle odesilatelovi paketu ICMP zprávu o nedostupnosti cíle.

V každém pravidle je specifikováno jaké vlastnosti musí paket splňovat, aby byl tímto pravidlem vyhodnocen. Základními možnostmi je filtrování podle čísla portu a zdrojové nebo cílové adresy paketu. Adresa může být zadána jménem nebo IP adresou. Dále lze uvést jméno použitého protokolu a síťového rozhraní, přes které chce paket projít. Pakety lze také filtrovat pomocí mac adresy zdroje odkud přišly. Užitečná může být také možnost nastavit limit počtu paketů, které odpovídaly danému pravidlu během určitého časového období. Pokud je tento limit překročen, budou všechny pakety odpovídající pravidlu zahozeny. Pakety spadající do řetězce OUTPUT lze filtrovat pomocí uid, gid, pid a sid, které paket vytvořily. V pravidlech může být také specifikován stav paketu. Rozlišují se čtyři stavy. Stav NEW říká, že paket navazuje nové spojení. Stav INVALID je nastaven u paketů, které nemohou být z nějakého důvodu identifikovány. Stav ESTABLISHED pro pakety patřící do existujícího navázaného spojení a stav RELATED pro pakety, které se spojením souvisí, ale nejsou jeho součástí.

Pokud paket projde celým řetězcem, aniž by odpovídal nějakému pravidlu, potom je vyhodnocen dle výchozí politiky řetězce. V praxi to znamená použití akce ACCEPT nebo DROP dle nastavené politiky. Lze také vytvářet nové vlastní řetězce, které je možné vkládat do řetězců výchozích.

Kapitola 3

Návrh

3.1 Analýza požadavků

Na základě zadání a specifikací vedoucího práce byly identifikovány základní požadavky na funkcionalitu výsledného modulu pro robota NAO. Dle zadání modul umožňuje připojení robota k WiFi sítí. Uživatel má možnost si tuto síť zvolit pomocí dotykových senzorů na hlavě robota. Robot pomocí hlasového výstupu komunikuje v českém nebo anglickém jazyce a reaguje na požadavky uživatele. WiFi síť je také zabezpečena firewallem.

- **Typ modulu**

  Výsledný modul je knihovna (lokální modul) načtená při startu NAOqi z inicializačního souboru autoload.ini.

- **Automatické připojení k síti po startu**

  Po startu modul automaticky připojí robota k dostupné WiFi síti s nejvyšší prioritou.

- **Manuální volba WiFi sítě**

  Uživatel si může zvolit síť ze seznamu dostupných sítí, které jsou nakonfigurované a je tak možné se k nim připojit. Požadované jsou zejména sítě FIT-1048 a eduroam, které jsou používány pro potřeby výuky.

- **Obnova po výpadku sítě**

  Pokud dojde k náhlému výpadku sítě, modul je schopen tento problém detekovat a připojení obnovit. Jestliže připojení není možné obnovit, nabídne modul uživateli možnost volby jiné dostupné sítě.

- **Pasivní a aktivní režim**

  Modul je schopen chodu v pasivním a aktivním režimu. V aktivním režimu běží modul na popředí a je možné volit sítě a jazyk. Pokud není aktivní režim ukončen uživatelem, přejde modul po určitém čase sám do pasivního režimu. V pasivním režimu běží modul na pozadí a neinteraguje na ovládací senzory, čímž je neblokuje pro využití jinými moduly. Mezi režimy je možné volně přepínat.

- **Hlasový výstup**

  Hlasový výstup informuje o všem podstatném, zejména o změnách stavu sítí a aktuálně vykonávaných požadavcích uživatele.

- **Vícejazyčnost**

  Veškerý hlasový výstup robota je podporován v českém a anglickém jazyce. Uživatel si může mezi těmito jazyky volit, buď při samotném běhu modulu nebo nastavením v konfiguračním souboru.

- **Načítání konfigurace z konfiguračního souboru**

  Konfigurace jednotlivých sítí spolu s preferovaným jazykem je uložena v konfiguračním souboru.

- **Nastavení firewallu pro danou síť**

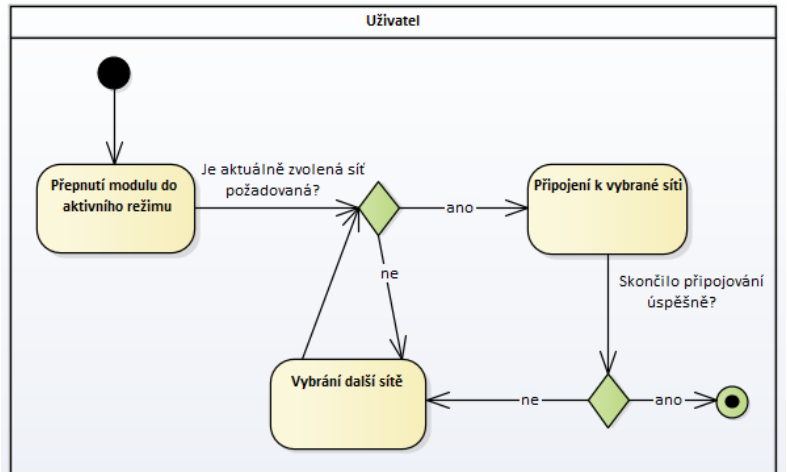
  Každé síti, ke které se robot připojí je nezbytné nastavit zabezpečení firewallem. Konfigurace firewallu se může pro jednotlivé sítě lišit podle toho, jací uživatelé k ní mají přístup a odkud.

3.2 Případy užití

Na základě požadavků z předcházející sekce jsou vytvořeny dva předpokládané nejčastější případy užití, které slouží pro přesnější představu fungování a používání modulu.

Připojení k uživatelem zvolené síti

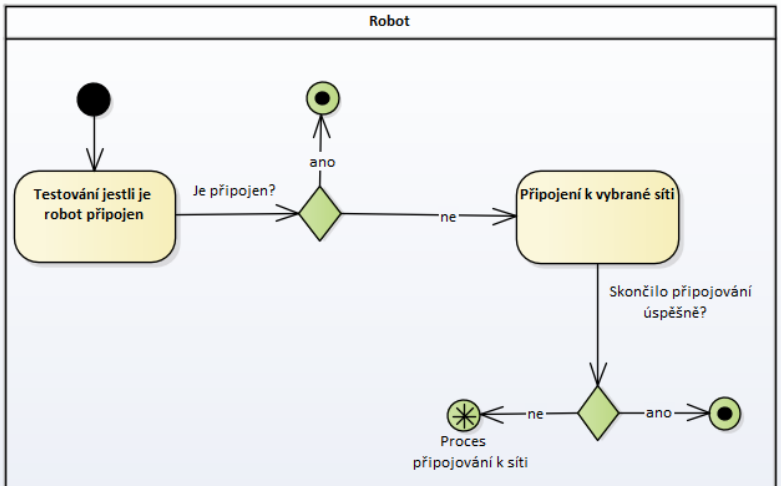
První případ užití popisuje postup připojení k uživatelem zvolené WiFi síti. Celý proces začíná přepnutím modulu do aktivního režimu. V aktivním režimu může uživatel připojit robota k aktuálně vybrané síti, nebo si zvolit další dostupnou síť v pořadí. Pokud připojení proběhne úspěšně, celý proces je ukončen. V opačném případě je uživatel vyzván, aby si vybral jinou síť a běh programu se vrací do bodu volby sítě. Diagram tohoto procesu zobrazuje obrázek 3.1.



Obrázek 3.1: Diagram procesu připojení k uživatelem zvolené síti

Obnova připojení po výpadku sítě

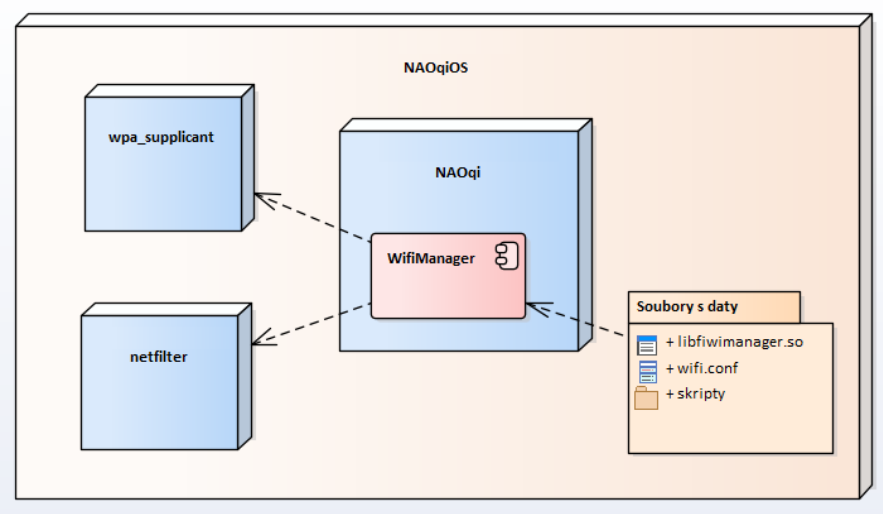
Druhý případ užití popisuje postup obnovy připojení po výpadku sítě. Proces začíná testováním, zda je robot připojen k WiFi síti. Jestliže je připojen, vše je v pořádku a není třeba v procesu pokračovat. Pokud robot není připojen, je provedeno připojení k naposledy vybrané síti. Jestliže připojování skončí úspěchem proces končí. Při neúspěchu je uživatel vyzván k vybrání jiné sítě a proces pokračuje již dříve popsaným procesem Připojení k uživatelem zvolené síti. Diagram tohoto procesu zobrazuje obrázek 3.2.



Obrázek 3.2: Diagram procesu obnovy připojení po výpadku sítě

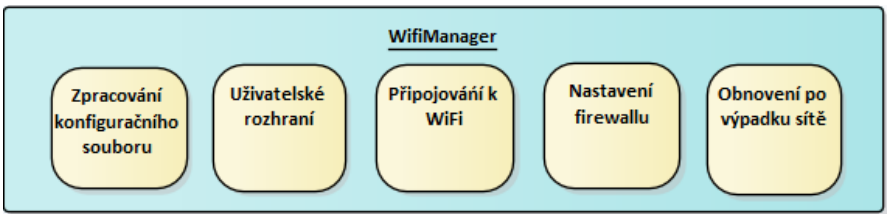
3.3 Návrh modulu

Modul je dle svého primárního účelu, tj. volba a připojování k WiFi sítím, pojmenován `WifiManager`. Integraci a nasazení modulu popisuje obrázek 3.3. Modul je navržen jako knihovna `libwifimanager.so` umístěná v adresáři s knihovnami /home/nao/lib/wifimanager v operačního systému robota. V tomto adresáři se také nachází konfigurační soubor a shellové skripty využívané modulem. Po spuštění procesu NAOqi, který řídí celého robota, je z knihovny vytvořena instance modulu `WifiManager`, která je součástí tohoto procesu. Modul pak může pomocí proxy objektů využívat služby ostatních lokálních modulů z frameworku NAOqi a spouštět shellové skripty potřebné pro konfiguraci a využívání služeb wpa_supplicantu a netfilteru.



Obrázek 3.3: Integrace a nasazení modulu WifiManager v systému robota NAO

Funkcionalita WifiManageru (obrázek 3.5) je rozdělena na pět základních oblastí, které vyplývají z na modul výše uvedených požadavků.



Obrázek 3.4: Základní funkcionalita WifiManageru

3.3.1 Zpracování konfiguračního souboru

Před samotným návrhem zpracování konfiguračního souboru je třeba říci, jaký je účel tohoto souboru, jaká data obsahuje a jak jsou tato data uložena. Konfigurační soubor slouží k načtení výchozí konfigurace po startu modulu. Obsahuje výchozí jazyk, kterým má robot mluvit a konfigurace jednotlivých WiFi sítí.

Struktura konfiguračního souboru

Při návrhu struktury konfiguračního souboru je kladen důraz především na snadnou tvorbu a modifikaci tohoto souboru. Předpokládá se, že soubor je modifikovaný administrátorem přímo z příkazové řádky, bez využití grafických editorů. Struktura proto musí být co nejjednodušší a tolerovat drobné nepřesnosti ve formátování souboru.

Struktura konfiguračního souboru je definována následujícími pravidly. Řádky začínající znakem # jsou považovány za komentáře až do konce řádku. Konfigurační soubor není citlivý na bíle znaky (space a tab) na začátku a konci řádku. Prázdné řádky nebo řádky obsahující pouze bíle znaky jsou přeskakovány. Každý záznam má formát parametr:hodnota a vyskytuje se maximálně jednou na řádku. Na pořadí zápisu jednotlivých parametru nezáleží. Konfigurace jednotlivých sítí jsou oddělené řetězcem `;;`. První validně zapsaná konfigurace WiFi je považována za výchozí a je použita při automatickém připojování po startu modulu.

Konfigurační soubor podporuje následující parametry:

- `language`: nastavuje výchozí jazyk modulu, hodnota může být *czech* pro češtinu nebo *english* pro angličtinu, v konfiguračním souboru musí být vždy uveden jako první parametr
- `type`: nastavuje jaky typ zabezpečení WiFi používá, hodnota může být *personal* pro sítě se zabezpečením WEP, WPA-PSK, WPA2 personal nebo *enterprise* pro sítě se zabezpečením WPA2 enterprise, jedná se o povinný parametr pro konfiguraci každé WiFi
- `ssid`: nastavuje identifikátor sítě WiFi, jedná se o povinný parametr pro konfiguraci každé WiFi
- `user`: nastavuje uživatelské jméno pod kterým se uživatel přihlašuje, hodnota je ve formátu uživatelskéjméno@realm, příklad: nao@fit.cvut.cz, jedná se o povinný parametr pro WiFi v režimu entreprise
- `password`: nastavuje klíč nebo heslo k WiFi, v konfiguračním souboru uveden pouze pokud je vyžadován pro autentizaci

- `namecz a nameen` : nastavuje jméno sítě, tak jak ji bude vyslovovat robot, vhodné zejména pro sítě jejichž ssid obsahuje zkratky nebo čísla, v konfiguračním souboru uveden volitelně

Příklad konfiguračního souboru:

```
#nastaveni jazyka
language: czech

type: personal
ssid: FIT-1048
namecz: fit deset čtyřicet osm
nameen: fit ten forty eight
password: heslo
;;

type: enterprise
ssid: eduroam
user: nao@fit.cvut.cz
password: heslo
;;
```

Zpracování konfiguračního souboru je zajištěno třídou `WifiLoader` vytvořenou speciálně pro tyto účely. Podrobný popis této třídy je v kapitole Implementace.

3.3.2 Uživatelské rozhraní

Jednou z nejdůležitějších věcí je návrh uživatelského rozhraní. Ovládání robota by mělo být co možná nejjednodušší. Uživatel potřebuje mít možnost přepnout robota do aktivního režimu, vybírat z dostupných WiFi sítí, zvolit vybranou síť a přepínat mezi jazyky modulu. K tomu se nejlépe hodí tlačítka a kapacitní senzory umístěné na těle robota. Díky využití modulu ALMemory z frameworku NAOqi lze zachytávat události vyvolané uživatelskou interakcí s těmito ovládacími prvky a pomocí metod modulu `WifiManager`, které implementují reakce na tyto události, vytvořit uživatelské rozhraní, řídící celý chod modulu.

Hlasový výstup

Robot nemá grafické rozhraní, pomocí kterého by si uživatel mohl ověřit, že robot reaguje na jeho pokyny. Součástí uživatelského rozhraní je proto i hlasový výstup robota, který tento nedostatek nahrazuje. S využitím modulu ALTextToSpeech z frameworku NAOqi je možné pomocí mluveného slova informovat o všech pro uživatele potřebných informacích.

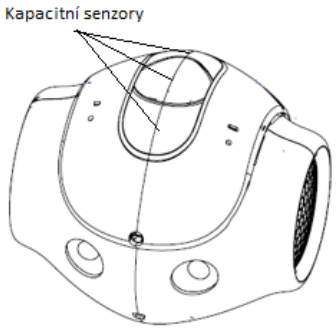
Přepnutí do aktivního režimu

Modul je dle požadavků navržen tak, aby umožňoval chod v pasivním a aktivním režimu. Tyto dva režimy jsou nezbytné, neboť robot disponuje pouze omezeným počtem tlačítek a senzorů pro manuální ovládání. Pokud mají dva moduly současně zaregistrované metody reagující na stejnou událost vyvolanou tlačítkem, reakce probíhají současně, což v konečném důsledku vede k nežádoucímu chování robota. Ve výchozím stavu je tedy modul v pasivním režimu, kdy nereaguje na žádné podněty od uživatele, kromě přepnutí do režimu aktivního. Pasivní režim tedy nechává ovládací prvky volné pro využití jinými moduly a zabraňuje tak nechtěným kolizím. V aktivním režimu modul reaguje na události vyvolané stiskem příslušných tlačítek. Pokud uživatel neukončí aktivní režim úspěšným připojením k síti, přejde modul sám po uplynutí dvouminutového časového intervalu do režimu pasivního o čemž uživatele hlasově informuje.

Pro přepnutí do aktivního režimu je využíváno tlačítko na hrudi robota. Konkrétně se jedná o tři stisknutí těsně za sebou. Takto vyvolaná událost musí být registrovaná pouze tímto modulem.

Výběr sítě

Výběr sítě je rozdělen na dvě části. Vybrání sítě ze seznamu dostupných nakonfigurovaných WiFi sítí a na zvolení této sítě pro připojení. K těmto účelům jsou použity tři kapacitní senzory umístěné na temeni hlavy robota (obrázek 3.5).



Obrázek 3.5: Umístění kapacitních senzorů na hlavě robota.(překresleno z dokumentace [1])

Po přepnutí do aktivního režimu, robot hlasově informuje o názvu aktuálně vybrané sítě. Dotykem předního senzoru lze vybrat následující síť v seznamu a dotykem zadního senzoru síť předcházející. Seznamem sítí tak lze procházet

oběma směry, což je výhodné zejména v případě nechtěného přeskočení požadované sítě, neboť kapacitní senzory jsou velmi citlivé a může dojít k dvojitému doteku. Dotykem prostředního senzoru je vybraná síť zvolena a začíná proces připojování.

Výběr jazyka

Poslední součástí uživatelského rozhraní je ovládání výběru jazyka. K přepínání mezi českým a anglickým jazykem by se nejvíce hodily kapacitní senzory umístěné na zápěstích robota. Tyto senzory však v době návrhu nebyly v pořádku a reagovaly velmi nespolehlivě. Z tohoto důvodu bylo k přepínání jazyka zvoleno stisknutí tlačítka (nárazníku) na špičce pravé nohy robota.

3.3.3 Připojování k WiFi síti

Pro připojování k WiFi sítím je využit modul ALConnectionManager z frameworku NAOqi. Tento modul poskytuje API umožňující připojování pouze k sítím, které nejsou zabezpečeny v režimu enterprise. Pro sítě v tomto režimu, konkrétně k síti eduroam, která se nachází na naší fakultě je proto využit nástroj wpa_supplicant. Při připojování je tedy nutné rozlišovat k jakému typu sítě se robot připojuje.

Při samotném připojování robot uživatele informuje, k jaké sítí se připojuje a jak toto snažení dopadlo. Modul si pamatuje poslední vybranou síť. Při příštím výběru sítě před připojováním je tato síť v seznamu dostupných sítí na prvním místě.

3.3.4 Obnova po výpadku sítě

Čas od času dochází k nečekanému výpadku připojení robota k síti. Ať už je tento výpadek způsoben čímkoli, je nezbytné automaticky detekovat tento problém a obnovit připojení, díky čemuž je uživatel jen minimálně ovlivněn při práci.

Detekce výpadku sítě je zajištěna vláknem vytvořeným při startu modulu `WifiManager`. Toto vlákno umožňuje každých 20 vteřin zkontrolovat, zda je robot připojen k síti. Testování stavu připojení probíhá pouze v okamžiku, kdy je modul v pasivním režimu. Pokud by bylo prováděno i v režimu aktivním, nebylo by možné rozlišit stav, kdy se robot právě odpojil a probíhá připojování k jiné uživatelem zvolené síti. V případě, že připojení není po testování v pořádku, vlákno se pokusí o opětovné připojení k této síti. Po neúspěšném pokusu o opětovné připojení je modul přepnut do aktivního režimu a uživatel je vyzván, aby si vybral síť jinou, neboť k současné se momentálně nelze připojit.

3.3.5 Nastavení firewallu

Připojením robota NAO k WiFi síti se otevírá prostor pro zneužití prostředků robota. Síťový provoz je proto třeba regulovat a stanovit jasná pravidla pro využívání jednotlivých sítí. K těmto účelům modul využívá nástroj iptables, pomocí kterého je nastaven firewall.

Základními hrozbami, které musí firewall řešit jsou:

- spouštění vzdálených modulů neoprávněnými uživateli
- práce více uživatelů současně

Spouštění vzdálených modulů není na robotovi nijak omezeno. Hrozí tedy, že si může kdokoli spustit svůj vlastní modul, aniž by o tom oprávněný uživatel věděl. Je proto nutností omezit přístup k robotovi z dané sítě pouze na oprávněné uživatele. Při práci více uživatelů současně je třeba dát pozor, aby nebyly spuštěny současně moduly vykonávající činnost, která by mohla ohrozit bezpečnost chodu robota. To lze v některých případech zajistit domluvou bezpečnostních pravidel pro používání robota, nebo zakázáním práce více uživatelů na úrovni firewallu.

Každé síti je možné nastavit pravidla firewallu přímo podle potřeb a využití sítě. Firewall je přizpůsoben pro sítě FIT-1048 a eduroam, které jsou využívány pro výuku. Pro ostatní sítě je v rámci maximální bezpečnosti použita výchozí konfigurace, povolující pouze protokol ssh na portu 22.

FIT-1048

Síť FIT-1048 je lokální síť umístěná v desátém patře budovy A Fakulty informačních technologií ČVUT. K této síti se mohou studenti připojit pouze z počítačů v učebně A-1048, kde probíhá výuka s robotem NAO. Identita těchto uživatelů je ověřena přihlášením se do systému počítačů. Všichni uživatelé jsou v jedné místnosti a tudíž je možné dohodnout společná pravidla používání robota v této učebně. Robot musí být vždy postaven na zemi, aby při špatném pohybu nedošlo k jeho pádu ze stolu. Uživatel při spouštění potenciálně nebezpečného modulu (většinou se jedná modul umožňující pohyb) informuje ostatní uživatele, že jde takovýto modul spouštět. Těmito pravidly lze dosáhnou kompromisu mezi bezpečností a použitelností práce s robotem.

Konfigurace firewallu tedy umožňuje komunikovat s robotem ze sítě FIT-1048 na všech portech za použití libovolných protokolů. Síťový provoz z jiných sítí je zakázán.

eduroam

Síť eduroam je globální síť jejíž uživatelé mohou být připojeni téměř z celého světa. Při výuce je tato síť využívána pro pro práci z vlastních počítačů studentů. Díky rozsahu pokrytí této sítě, uživatelé nemusí být v vůbec v místnosti s robotem. Nelze tak domluvit a vyžadovat pravidla používání, protože k síti může být připojen kdokoli. Je tedy nutné pomocí firewallu omezit přístup uživatelů z této sítě. Povolen je tedy jen protokol ssh, pomocí kterého se můžou učitelé a vybraná skupina studentů přihlásit do operačního systému robota. Takovýto uživatel, který má navíc přidělena administrátorská práva, může pomocí vytvořeného skriptu povolit přístup k robotovi z maximálně jedné IP adresy sítě eduroam současně.

Konfigurace firewallu tedy umožňuje komunikovat s robotem ze sítě eduroam protokolem ssh na portu 22. Pomocí vytvořeného skriptu v operačním systému robota pak lze povolit veškerou komunikaci z konkrétní IP adresy této sítě. Síťový provoz z jiných sítí je zakázán.

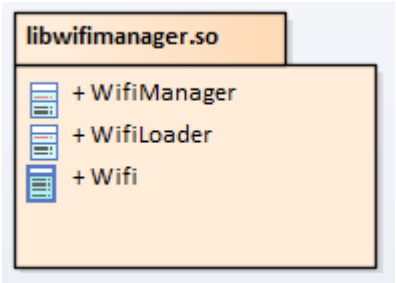
Kapitola 4

Implementace

Prvním krokem při tvorbě modulu je výběr programovacího jazyka. Pro tvorbu nových lokálních modulů (knihoven) za použití frameworku NAOqi jsou podporovány jazyky Python a C++. Jazyk Python se používá zejména pro programování chování robota v prostředí Choregraphe. Pro implementaci byl proto zvolen jazyk C++, který je pro účely tvorby nových modulů vhodnější. Ke tvorbě modulu se nabízí v co možná největší míře využívat frameworku NAOqi. Framework poskytuje API pro tvorbu modulů, která jsou při návrhu a implementaci využita.

4.1 Struktura modulu

Požadovaný modul je navržen jako lokální modul. Výsledkem kompilace je knihovna `libwifimaneger.so`, která je přidána do inicializačního souboru autoload.ini v operačním systému robota. Po zapnutí robota je tedy modul načten a spuštěn v procesu NAOqi. Program využívá metod asynchronního programování, neboť tok běhu programu je řízen událostmi. Modul se skládá ze tříd `WifiManager, WifiLoader` a struktury `Wifi` jak je patrné z obrázku 4.1.



Obrázek 4.1: Třídy tvořící knihovnu libwifimanager.so

WifiManager

Třída `WifiManager` je navržena dle návrhového vzoru pro tvorbu nových lokálních modulů uvedeného v dokumentaci SoftBank Robotics [1]. Je tedy potomkem třídy ALModule z frameworku NAOqi. Deklarace třídy je uložena v souboru wifimanager.h a implementace v souboru wifimanager.cpp. Účelem této třídy je implementovat metody využívající služeb ostatních lokálních modulů NAOqi. V konstruktoru třídy jsou zaregistrovány metody, které je potom možné pomocí proxy volat z jiných modulů. Následující příklad kódu zobrazuje registraci metody `isOnline`.

```
functionName("isOnline", getName(),
"Check if robot is connected to WiFi.");
BIND_METHOD(WifiManager::isOnline);
```

Důležitou metodou třídy je metoda `init`, která se provede vždy ihned po vytvoření instance modulu. Po doběhnutí této metody je běh programu řízen událostmi. Metoda nejprve vytvoří instanci třídy `WifiLoader wifi` zodpovědnou za načtení dat z konfiguračního souboru a následně inicializuje členské proměnné na výchozí hodnoty. V metodě `init` jsou vytvořeny proxy objekty na moduly procesu NAOqi `AL::ALMemoryProxy fMemoryProxy`, umožňující práci s pamětí robota, `AL::ALTextToSpeechProxy fTtsProxy`, zajišťující hlasový výstup a proxy `AL::ALConnectionManagerProxy fCmanProxy` sloužící pro volání metod potřebných pro připojování k WiFi síti.

Wifi

Struktura `Wifi` reprezentuje WiFi síť. Její využití je zejména pro uchování a práci s daty načtenými z konfiguračního souboru, čemuž odpovídají i její členské proměnné. Deklarace struktury je uložena v souboru wifiloader.h.

```
struct Wifi
{
  string w_type;
  string w_ssid;
  string w_user;
  string w_password;
  string w_id;
  string w_namecz;
  string w_nameen;
};
```

WifiLoader

Třída `WifiLoader` poskytuje API pro načtení dat z konfiguračního souboru a jejich následné zpracování. Deklarace třídy je uložena v souboru wifiloader.h a implementace v souboru wifiloader.cpp.

Členská proměnná `language`, reprezentuje zvolený jazyk modulu. Do vektoru `configured_wifi` jsou ukládány konfigurace jednotlivých sítí načtené z konfiguračního souboru. Vektor `found_wifi`, obsahuje WiFi sítě, které jsou dostupné během skenování síťového rozhraní. Poslední proměnnou je vektor `available_wifi`, který obsahuje průnik předchozích dvou vektorů, tj. sítě ke kterým se lze připojit.

Načtení a zpracování konfiguračního souboru /home/nao/lib/wifimanager/wifi.conf zajišťuje metoda `loadConfig`. Soubor je čten po řádcích, které jsou parsovány na jednotlivé parametry a jejich hodnoty. Metoda `clearBlank`, umožňuje odstranění bílých znaků (mezer a tabelátorů) ze začátku a konce zadaného řetězce.

```
ifs.open ("/home/nao/lib/wifimanager/wifi.conf", ifstream::in);
    while (ifs.good())
    {
      getline (ifs,line);
      clearBlank(line);
      if(line[0]!='#' && line.size()!=0)
      {
        breakpoint=line.find_first_of(":");
        parameter = line.substr(0,breakpoint);
        value = line.substr(breakpoint+1);
        clearBlank(parameter);
        clearBlank(value);
        ...
```

Výsledné konfigurace sítí jsou ukládány do vektoru `configured_wifi`. Na závěr je ještě provedeno testování zda všechny nakonfigurované WiFi sítě, uložené ve vektoru, obsahují alespoň minimální potřebnou konfiguraci. Pokud během zpracování dojde k nalezení neplatných dat, tato data jsou přeskočena a metoda vrací řetězec obsahující zprávu o daném problému, která je později zapsána do errorlogu NAOqi.

4.2 Reakce na události

Jak už bylo dříve řečeno chod modulu je řízen událostmi. Pro práci s událostmi je nutné registrovat metody, které budou zavolány pokud k události dojde. Třída `WifiManager` implementuje následující seznam takovýchto metod:

`onTripleClickOccurred`

Tato metoda je zavolána po trojkliku uživatele na hrudní tlačítko robota. Metoda slouží k přepnutí robota do aktivního režimu nastavením proměnné `active` na hodnotu true a registrací metod reagujících na stisk tlačítek robota. V pasivním režimu byly v metodě `init` registrovány pouze metody `onNetworkServiceInputRequired` a `onTripleClickOccurred`, které mohou být volány v obou režimech.

```
...
active=true;
fMemoryProxy.subscribeToEvent("FrontTactilTouched","WifiManager",
"onFrontTactilTouched");
fMemoryProxy.subscribeToEvent("MiddleTactilTouched","WifiManager",
"onMiddleTactilTouched");
fMemoryProxy.subscribeToEvent("RearTactilTouched","WifiManager",
"onRearTactilTouched");
fMemoryProxy.subscribeToEvent("RightBumperPressed","WifiManager",
"onRightBumperPressed");
...
```

Následně je pomocí metody `fTtsProxy.say` uživatel hlasově informován o změně režimu modulu a aktuálně vybrané síti.

`onFrontTactilTouched`

Tato metoda reaguje na dotek předního kapacitního senzoru na hlavě robota. Metoda slouží k nastavení aktuálně vybrané WiFi sítě v proměnné `selected`, která reprezentuje pozici sítě ve vektoru `available_wifi`. Proměnná je inkrementována o jedničku čímž je vybrána následující síť. Uživatel je hlasově informován o názvu nově vybrané sítě. Metody reaugující na události je také nutné zamykat viz:

```
float fState;
    AL::ALCriticalSection section(fCallbackMutex);
    fState =  fMemoryProxy.getData("RearTactilTouched");
    if (fState  > 0.5f)
    {
      return;
    }
```

`onRearTactilTouched`

Metoda `onRearTactilTouched` je analogií metody předešlé, s tím rozdílem, že reaguje na dotek zadního kapacitního senzoru na hlavě robota a vybírá síť předcházející.

`onRightBumperPressed`

Tato metoda reaguje na zmačknutí nárazníku (bumperu) na pravé noze robota. Jejím účelem je změnit aktuálně používaný jazyk modulu. Pokud je tedy před zmačknutím vybrána čeština, jazyk je změněn na angličtinu a opačně. Změna jazyku je provedena pomocí metody `fTtsProxy.setLanguage`.

`onMiddleTactilTouched`

Metoda `onMiddleTactilTouched` reaguje na dotek prostředního kapacitního senzoru na hlavě robota. Cílem této metody je zahájit připojování k aktuálně vybrané síti. Pomocí metody `fTtsProxy.say` je uživatel hlasově informován o výsledku připojování. Pokud připojování skončí úspěšně, modul je přepnut do pasivního režimu nastavením proměnné `active` na hodnotu false a provedením odregistrování metod reagujících na stisk tlačítek robota.

`onNetworkServiceInputRequired`

Tato metoda reaguje na událost vyvolanou metodou `fCmanProxy.connect`, která pro své úspěšné dokončení vyžaduje zaslat objekt `AL::ALValue` s požadovanými autentikačními údaji. Způsob tohoto předání zobrazuje následující část kódu:

```
AL::ALValue request = fMemoryProxy.getData("NetworkServiceInputRequired");
  int s=request.getSize();
    AL::ALValue value1;
    AL::ALValue value2;
    AL::ALValue reply;
    value1.arrayPush("ServiceId");
    value1.arrayPush(wifi.getId(selected));
    reply.arrayPush(value1);
    value2.arrayPush("Passphrase");
    value2.arrayPush(wifi.getPassword(selected));
    reply.arrayPush(value2);
    try
    {
      fCmanProxy.setServiceInput(reply);
    }
    ...
```

4.3 Proces připojování

Pokud není uvedeno jinak, předpokládáme v této sekci užití namespacu WifiManager. Před každým připojením nebo výběrem jiné sítě je nutné zavolat metodu `getAvailibleWifi`, která pomocí metody `fCmanProxy.services` provede skenování síťového rozhraní a vrátí objekt reprezentující nalezené sítě. Tento objekt je pak zpracován metodou `wifi.loadAvailable`. Po setřídění vektorů `configured_wifi` a `found_wifi` metodou `wifi.sortAvailable` je seznam sítí dostupných pro připojení aktuální ve vektoru `available_wifi.`V tomto aktualizovaném vektoru může být jiný počet sítí, než ve vektoru původním. Je proto nutné nastavit proměnnou `selected`na správnou hodnotu zavoláním metody `wifi.findSelected`s parametrem `connectedName,`který reprezentuje ssid naposledy vybrané sítě. Pokud už hledaná síť není dostupná, je nastavena proměnná `selected`na hodnotu 0, což je pozice sítě s aktuálně nejvyšší preferencí.

Před zahájením připojování je nutné zajistit odpojení předchozího připojení. To zajišťuje skript `home/nao/lib/wifimanager/clearsupplicant.sh` volaný z metody `clearSupplicant`.

```
#!/bin/bash
IFS=$'\n'
config=‘wpa_cli list_networks | tail -n +3 | cut -f 1‘
for item in $config
do
   wpa_cli disconnect
   wpa_cli disable_network $item
   wpa_cli remove_network $item
done
found=‘connman services | grep "^\* A.*{ wifi" \
| cut -d "{" -f2 | cut -d " " -f2‘
for item in $found
do
   connman remove $item
done
dhclient -r wlan0;
killall dhclient;
```

Skript odstraňuje všechny nakonfigurované WiFi sítě z wpa_supplicantu, odstraní konfiguraci sítí z connmanu a odebere IP adresu z rozhraní wlan0 přidělenou pomocí dhclientu. Ukončuje také všechny staré procesy dhclientu, které zůstávají viset v systému.

Připojování k Wifi zajišťuje metoda `connectWifi`, která pro připojení k sítím typu personal využívá metodu `fCmanProxy.connect`. Pokud jsou pro připojení požadovány nějaké povinné autentizační údaje, metoda vyvolá událost NetworkServiceInputRequired, na kterou je třeba odpovědět. Pro sítě se zabezpečením typu enterprise je využívána metoda `connectEnterprise`. Tato metoda pomocí systémového volání spouští skript /home/nao/lib/wifimanager/eduroam.sh.

```
string runEnterprise="sudo /home/nao/lib/wifimanager/eduroam.sh "
+ wifi.getSsid(selected) + " " + wifi.getUser(selected) + " " +
wifi.getPassword(selected);
system(runEnterprise.c_str());
```

Skript pomocí wpa_cli nakonfiguruje wpa_supplicant pro připojení k síti eduroam. Protože je v místě testování slabý signál, je někdy nutné provést více pokusů o připojení. Následující ukázka skriptu je provedena až třikrát, pokud je to nezbytné. Po navázání připojení je síťovému rozhraní wlan0 přidělena Ip adresa pomocí dhclientu.

```
NET=‘wpa_cli add_network | tail -1‘
wpa_cli set_network $NET ssid \""$1"\";
wpa_cli set_network $NET key_mgmt WPA-EAP;
wpa_cli set_network $NET eap PEAP;
wpa_cli set_network $NET phase2 \"auth=MSCHAPV2\";
wpa_cli set_network $NET pairwise CCMP;
wpa_cli set_network $NET group CCMP TKIP;
wpa_cli set_network $NET domain_suffix_match \"radius.fit.cvut.cz\"
wpa_cli set_network $NET ca_cert \
\"/home/nao/lib/wifimanager/chain_TERENA_eScience_SSL_CA_2.pem\";
wpa_cli set_network $NET identity \""$2"\";
wpa_cli set_network $NET password \""$3"\";
wpa_cli select_network $NET;
for (( c=1; c<=4; c++ ))
do
    wpa_cli status | grep -w wpa_state=COMPLETED
    if [ $? -eq 0 ]; then
        dhclient wlan0;
        exit 0
    fi
    sleep 2;
done
...
```

Na závěr metody `connectWifi` je zavolána metoda `setFirewall`, která pomocí systémového volání spouští skript /home/nao/lib/wifimanager/firewall.sh konfigurující firewall pro připojovanou síť.

V návaznosti na tento skript byl implementován skript `allowIP.sh` dostupný z CD v příloze, který administrátorovi slouží k dodatečnému povolování přístupu k robotovi z jedné parametrem zadané IP adresy.

Metoda `isOnline`, umožňuje zjistit aktuální stav připojení. Skript /home/nao/lib/wifimanager/isonline.sh, který tato metoda využívá není spouštěn pomocí bežného systémového volání, ale pomocí metody `popen`, umožňující zpracovat standardní výstup skriptu v těle metody `isOnline`.

```
FILE *in;
  char buff[1024];
  if(!(in = popen("sudo /home/nao/lib/wifimanager/isonline.sh", "r")))
  {
     qiLogError("WifiManager")
     << "Connection to enterprise failed." << endl;
  }
  fgets(buff, sizeof(buff), in);
  pclose(in);
  if (strcmp(buff,"1") == 0) return true;
  else return false;
```

4.4 Kontrolní vlákno

V konstruktoru třídy `WifiManager` je vytvořeno vlákno `backgroundThread` ve kterém běží metoda `checkStatus`. Dokud není v destruktoru třídy nastavena proměnná `turnoff` na hodnotu true a zavolána metoda `backgroundThread.join`, provádí se vykonávání metody v nekonečné smyčce, jejíž iterace se opakuje každých 20 vteřin, což je doba po kterou vlákno vždy spí.

```
while (!turnoff)
  {
    boost::this_thread::sleep_for(boost::chrono::seconds(20));
...
```

Metoda `checkStatus` má dva hlavní účely. V pasivním režimu modulu kontroluje každých 20 vteřin, zda nedošlo k výpadku sítě. Při výpadku sítě, metoda hlasově informuje o výpadku sítě a pokusí se o opětovné připojení. Pokud se připojení obnovit nepodařilo, přepne modul do aktivního režimu a požádá uživatele, aby si zvolil jinou síť. Při výpadku sítě jsou rozlišovány dva případy. V prvním byla síť odpojena a je třeba zavolat metodu `connectWifi` pro opětovné připojení. Druhý případ nastává, pokud je síť stále připojena, ale

rozhraní wlan0 nemá nastavenu IP adresu. V tomto případě je rychlejší pouze dhclientem znovu přidělit IP adresu. To zařídí metoda `isDHCP`, která spustí skript home/nao/lib/wifimanager/isDHCP.sh

V aktivním režimu modulu, proměnná `active` je nastavena na hodnotu true, slouží metoda k přepnutí modulu do režimu pasivního. To se stane pouze v případě, že uživatel s robotem více jak 2 minuty nekomunikoval, tzn. nezmáčkl žádné z ovládacích tlačítek. Implementace tohoto timeoutu je velice snadná. Probudí-li se vlákno v aktivním režimu, inkrementuje si proměnnou `idle` o jedna. Pokud uživatel mezi tím zmačkne nějaké tlačítko, je v metodě reagující na tuto událost nastavena hodnota `idle` zpět na nulu. Při dosažení větší hodnoty než 6, jsou provedeny již výše popsané kroky pro přepnutí do pasivního režimu.

Kapitola 5

Testování

Testování funkcionality modulu probíhalo po celou dobu návrhu a vývoje přímo na robotovi NAO. Modul byl nejprve spouštěn jako binární soubor z operačního systému robota. Tento způsob testování umožňoval jednodušší ladění v raných fázích vývoje, neboť testovací výpisy z běhu modulu bylo možno zobrazovat přímo do terminálu. Tento způsob testování však narazil na hranici svých možností v okamžiku, kdy byl robot odpojen od ethernetu, čímž přestal fungovat broker, který modul využíval. Později již byl proto modul kompilován jako knihovna a spouštěn přímo z procesu NAOqi. Tím se problém s brokerem vyřešil, neboť všechny moduly v procesu NAOqi mají broker společný. Nepřehledným se naopak stalo ladění, protože bylo možné zobrazovat pouze výpisy ze všech modulu NAOqi současně. Výsledkem tohoto testování byl odladěný modul, připravený pro použití v praxi.

5.0.1 Specifika prostředí

Připojování k síti eduroam pomocí wpa_supplicantu odhalilo hned několik specifik operačního systému robota, na které je třeba si dát pozor. Po nastavení konfigurace sítě eduroam, která byla úspěšně otestována na jiném stroji, hlásil wpa_cli při pokusu o připojení chybu *err='Server used client certificate'*. Tento problém je způsoben chybou ve wpa_supplicantu verze 2.1 devel, který je dostupný v operačním systému robota. Řešením je nainstalovat novější verzi wpa_supplicantu. Ani to však nebylo úplně jednoduché, neboť systém nebyl již několik let aktualizován a nelze tak použít balíčkovací systém Portage. Novější verze tak musela být zkompilována ručně.

Aby problémům s připojením k eduroamu nebyl konec, je třeba zajistit, aby čas modifikace souboru s certifikátem pro ověření identity radius serveru byl pozdější, než čas zahájení platnosti certifikátu. Robot totiž nemá v hlavě nabitou baterii a nelze tak nastavit hardwarový čas na aktuální hodnotu. Všechny nově vytvořené soubory tak mají staré datum.

5.0.2 Navržená zlepšení

Při testování hlasového výstupu robota byla identifikována potřeba zlepšení výslovnosti názvů WiFi sítí. Dle původního návrhu robot vyslovoval jako název sítě její SSID. V případě zkratek nebo názvů netvořených slovy však nebyl robot schopen tato jména uživatelovu uchu přijatelně vyslovit. Řešení spočívá v rozšíření možností konfiguračního souboru o volitelné parametry namecz a nameen, které slouží jako výslovnost jména sítě. Příkladem nechť je síť FIT-1048 kterou robot vyslovoval jako „Fit tisíc čtyřicet osm“, ale uživatelé jí znají pod názvem „Fit deset čtyřicet osm“.

Závěr

Výstupem bakalářské práce je modul určený pro robota NAO, umožňující správu WiFi připojení. V teoretické části práce byly prozkoumány možnosti a dostupné technologie potřebné pro realizaci modulu. Byla také provedena rešerše existujících řešení, jejíž výsledky byly zohledněny pří návrhu modulu.

Na základě požadavků a případů užití byl navrhnut a implementován lokální modul pro robota NAO. Modul je uživatelem ovladatelný pomocí tlačítek a senzorů na těle robota. Umožňuje uživateli volbu Wifi sítě ze seznamu dostupných sítí, hlasově informuje o změně stavu sítě a výsledku připojování. Hlasový výstup je plně podporován jak v českém, tak v anglickém jazyce a mezi jazyky lze libovolně přepínat. Při náhlém výpadku sítě je modul schopen automatické obnovy připojení. Pro načítání konfigurací jednotlivých sítí je využíván konfigurační soubor.

Na základě analýzy bezpečnostních rizik připojení robota k WiFi síti bylo navrženo řešení přinášející přijatelný kompromis mezi bezpečností a použitelností. Toto řešení je každé WiFi síti vytvořeno na míru, podle typu a způsobu využívání. Skládá se z doporučení pravidel pro používání robota v této síti a nastavení firewallu modulem při připojování.

Funkčnost implementovaného modulu byla otestována přímo na robotovi NAO. V současné době je modul začleněn do procesu NAOqi a připraven k použití v běžném provozu.

Literatura

[1] Aldebaran 2.1.4.13 documentation. [online], ©2017, [cit. 2017-4-15]. Dostupné z: http://doc.aldebaran.com/2-1/index.html

[2] Bradley Mitchell: What is a WEP Key in Wi-Fi Networking? [online], March 10, 2017, [cit. 2017-5-13]. Dostupné z: https://www.lifewire.com/what-is-a-wep-key-818305

[3] What is Wi-Fi Protected Access Pre-Shared Key (WPA-PSK)? - Definition from Techopedia. [online], [cit. 2017-5-13]. Dostupné z: https://www.techopedia.com/definition/22921/wi-fi-protected-access-pre-shared-key-wpa-psk

[4] Wi-Fi Protected Access 2 (WPA 2) Configuration Example - Cisco. [online], January 21, 2008, [cit. 2017-5-13]. Dostupné z: http://www.cisco.com/c/en/us/support/docs/wireless-mobility/wireless-lan-wlan/67134-wpa2-config.html

[5] Configuring RADIUS Authentication with WPA2-Enterprise. [online], 2017-04-17T19:26:42Z, [cit. 2017-5-14]. Dostupné z: https://documentation.meraki.com/MR/Encryption_and_Authentication/Configuring_RADIUS_Authentication_with_WPA2-Enterprise

[6] Wpa_supplicant(8) - Linux man page. [online], [cit. 2017-6-8]. Dostupné z: https://linux.die.net/man/8/wpa_supplicant

[7] Harald Welte, Pablo Neira Ayuso: The netfilter.org project. [online], ©1999-2014, [cit. 2017-5-13]. Dostupné z: https://www.netfilter.org/

Příloha A

Seznam použitých zkratek

BI-ZIVS Základy inteligentních vestavných systémů

MI-IVS Inteligentní vestavné systémy

RAM Random-access memory

LED Light-Emitting Diode

USB Universal Serial Bus

SOAP Simple Object Access Protocol

XML eXtensible Markup Language

API Application Programming Interface

WEP Wired Equivalent Privacy

WPA-PSK Wi-Fi Protected Access Pre-Shared Key

ICMP The Internet Control Message Protocol

TKIP Temporal Key Integrity Protocol

CCMP Counter Mode Cipher Block Chaining Message Authentication Code Protocol

EAP Extensible Authentication Protocol

PEAP Protected Extensible Authentication Protocol

Příloha B

Uživatelská příručka

B.1 Kompilace modulu

Kompilace modulu byla prováděna na 64-bitovém operačním sytému Ubuntu 14.04.5. Na systému byl nainstalovaný CMake verze 3.7.0, gcc verze 4.8.4 a qibuild verze 3.11.6.

Pomocí příkazu
`qitoolchain create robot /path/to/SDKfolder/naoqi-sdk/toolchain.xml`
je vytvořen toolchain robot. Příkaz `qibuild init` inicializuje adresář ve kterém je zavolán jako worktree. V tomto adresáři lze příkazem `qisrc create jméno` vytvořit adresáře s jednotlivými projekty. Příkazem `qibuild configure -c robot` se v adresari projektu provede konfigurace a příkazem `qibuild make -c robot` kompilace modulu.

B.2 Nasazení modulu

Kořenový adresář modulu je umístěn v `/home/nao/lib/wifimanager.` Do tohoto adresáře je třeba zkopírovat obsah zazipovaného adresáře `wifimanager.zip` z CD. Adresář by měl potom obsahovat knihovnu `libwifianager.so`, podpůrné skripty a certifikát radius serveru. Následujícími příkazy je pak třeba nastavit správná práva všem souborům:

```
cd /home/nao/lib/wifimanager
chown nao:nao *
chmod 755 *.sh *.pem
chmod 700 wifi.conf
```

POZOR !!!. Soubor s certifikátem `chain\_TERENA\_eScience\_SSL\_CA\_2.pem` musí mít nastavený čas modifikace po 9.8.2014, kdy začíná doba jeho platnosti.

Pro spouštění skriptů pod privilegovaným uživatelem je třeba přidat do soubor `/etc/sudoers` příkazem `sudo visudo` následují nastavení.

```
nao ALL= NOPASSWD: /home/nao/lib/wifimanager/clearsupplicant.sh
nao ALL= NOPASSWD: /home/nao/lib/wifimanager/eduroam.sh
nao ALL= NOPASSWD: /home/nao/lib/wifimanager/firewall.sh
nao ALL= NOPASSWD: /home/nao/lib/wifimanager/isonline.sh
nao ALL= NOPASSWD: /home/nao/lib/wifimanager/isDHCP.sh
```

Dále je třeba přidat cestu ke knihovně `libwifianager.so` do souboru `/etc/naoqi/autoload.ini`.

Poslední věcí, kterou je nutno udělat je zprovoznit wpa_supplicant. Snažší část zahrnuje přidání řádku
`Exec=/usr/sbin/wpa_supplicant -u -O /var/run/wpa_supplicant`
do souboru
`/usr/share/dbus-1/system-services/fi.w1.wpa_supplicant1.service`. Obtížnější část spočívá v nahrazení wpa_supplicantu 2.1 devel verzí 2.5, jejíž zdrojový kód lze najít na CD v adresáři tools v souboru `wpa_supplicant-master.zip`. Wpa_supplicant je zkompilován dle pokynů uvedených v souboru README za využití přednastavené konfigurace ze souboru `config` umístěné na CD. Pak už jen stačí překopírovat nově zkompilované binární soubory `wpa_cli`, `wpa_supplicant` a `wpa_passphrase` místo souborů původních.

B.3 Ovládání modulu

Ovládání modulu je velmi jednoduché:

- **trojklik na hrudním tlačítku:** přepnutí modulu do aktivního režimu
- **přední kapacitní senzor na hlavě:** výběr následující sítě
- **zadní kapacitní senzor na hlavě:** výběr předcházející sítě
- **prostřední kapacitní senzor na hlavě:** zvolení sítě a zahájení připojování
- **nárazník (bumper) na pravé noze:** změna jazyka modulu

Konfigurační soubor a skripty

- **wifi.conf**

  Hlavní konfigurační soubor, slouží k nastavení výchozího jazyka modulu a konfiguraci jednotlivých WiFi sítí. Pravidla pro editaci jsou popsána uvnitř souboru.

- **allowIP.sh**

  Tento skript slouží k povolení konkrétní IP adresy v síti eduroam. Adresa je při spuštění zadána jako parametr skriptu. Vyžaduje spouštění pod uživatelem root.

- **firewall.conf**

  Skript umožňující konfiguraci firewallu podle ssid WiFi sítě, ke které je robot připojen. Pravidla pro novou síť se zapisují do bloku case.

- **eduroam.sh**

  Skript umožňuje připojení k síti eduroam. Editace je nutná pouze v případě, že došlo ke změně v konfiguraci této sítě a je třeba předávat wpa_cli jiné hodnoty.

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `wifimanager.zip` ..... adresář se soubory modulu
- `refman.pdf` ..... programátorská dokumentace
- `allowIP.sh` ..... skript pro konfiguraci firewallu
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `BP_Jelinek_Tomas_2017.pdf` ..... text práce ve formátu PDF
- `tools`
  - `wpa_supplicant-master.zip` ..... wpa_supplicant 2.5
  - `config` ..... konfigurační soubor pro kompilaci wpa_supplicantu