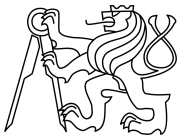
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	BACnet klient pro mobilní platformu Android
Student:	Bc. Petr Kubišta
Vedoucí:	Ing. Miroslav Balík, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Vytvořte klientskou aplikaci pro monitorování a ovládání systémů řízení vytápění, ventilace, klimatizace a dalších zařízení v oblasti automatizace budov používajících komunikační protokol BACnet.
1. Analyzujte existující knihovny implementující protokol BACnet pro různé platformy a ověřte jejich využitelnost pro platformu Android.
2. Implementujte protokol BACnet/IP na platformě Android. Implementujte podporu základních typů BACnet objektů a služeb. Analyzujte možnost implementace komplexních objektů a služeb a protokolu BACnet MS/TP.
3. Navrhněte a implementujte uživatelské rozhraní umožňující monitoring a ovládání BACnet zařízení a jejich objektů. Zaměřte se na snadné a přívětivé ovládání pro techniky správy budov i běžné uživatele.
4. Navrhněte a implementujte vhodný způsob konfigurace databáze BACnet zařízení a objektů ve vazbě na UI.
5. Řešení podrobte testům stability implementace protokolu BACnet a funkčnosti uživatelského rozhraní. Vyhodnoťte výsledky testování.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 29. prosince 2015

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práce

Android aplikace pro ovládání BACnet® zařízení

Bc. Petr Kubišta

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

9. května 2016

Poděkování

V první řadě bych chtěl poděkovat svým rodičům za neustálou podporu v průběhu celého studia. Další díky patří mému vedoucímu, panu proděkanovi Ing. Miroslavovi Balíkovi, Ph.D., za jeho rady a konstruktivní kritiku v průběhu vypracování této práce. A v neposlední řadě bych chtěl také poděkovat všem svým přátelům, kteří mi zpříjemnili průchod studiem.

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 9. května 2016 .....

Odkaz na tuto práci

Kubišta, Petr. *Android aplikace pro ovládání BACnet® zařízení.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

BACdroid je mobilní aplikace a přidružený konfigurační systém, sloužící k snadnému ovládání zařízení implementující protokol BACnet®. Lze ji využít v domácnostech i inteligentních budovách, jejichž technologie jsou ovládány řídícími systémy podporujícími komunikační protokol BACnet®.

Klíčová slova BACnet®, BACdroid, Mobilní aplikace, Ovládání na dálku, Řídící systém, Vzduchotechnika, Konfigurace

Abstract

BACdroid is a mobile application and an associated configuration system, used for effortless management of devices implementing BACnet® protocol. It can be used in households and smart buildings, whose equipment is managed by controllers supporting BACnet® communication protocol.

Keywords BACnet®, BACdroid, Mobile application, Remote control, Control System, Ventilation, Configuration

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Členění práce

Účelem této práce je vytvoření mobilní aplikace a konfiguračního systému pro komunikaci se zařízeními implementujícími protokol BACnet®. Nejprve bude provedena analýza standardu BACnet®, knihoven, které tento protokol implementují, a podobných aplikací, které již existují.

Následně bude proveden návrh aplikace a přidruženého systému, a na základě tohoto návrhu bude aplikace implementována pro mobilní platformu Android™.

Využitá knihovna i vzniklá aplikace bude otestována při komunikaci se skutečnými BACnet® zařízeními. Aplikace bude následně uživatelsky otestována s ohledem na navržené uživatelské rozhraní.

Na závěr budou zmíněny možnosti dalšího vývoje a vylepšení systému, který vznikne při vypracování této práce.

Podíl autora na práci

Autor práce zpracuje aplikaci i přidružený systém vlastnoručně, jediným prvkem, který nebude prací autora, bude využitá knihovna přímo komunikující se zařízeními BACnet®. Aplikace včetně uživatelského rozhraní bude od základu vyvinuta za použití Android studia, vývojového nástroje určeného k tvorbě Android aplikací.

Protokol BACnet®

Protokol BACnet®, neboli Data Communication Protocol for Building Automation and Control Networks (datový komunikační protokol pro automatizaci budov a řídící sítě), byl vyvinut pod záštitou American Society of Heating, Refrigerating and Air-Conditioning Engineers (Americké společnosti inženýrů

vytápění, chlazení a klimatizace), zkráceně ASHRAE. Jde o americký národní standard, evropský standard, národní standard více než 30 zemí, a globální ISO standard.

Pro účel této aplikace bude využit konkrétně BACnet/IP, který je pro zařízení Android nejvhodnější. Je sice možné přistupovat ze zařízení Android přímo na sériovou linku, ale tento způsob je poměrně obtížný.

Zařízení implementující BACnet® protokol se v praxi sdružují do sítí. V případě BACnet/IP zná každé zařízení v síti svoji IP adresu.

Pro komunikaci mezi zařízeními se používá protokol UDP, který umožňuje zasílat zprávy typu broadcast (jeden-mnoha). Mimo nich využívá BACnet/IP ještě tzv. unicast, zprávu z jednoho zařízení druhému.

BACnet® jako protokol definuje pouze způsob komunikace po síti, nikoliv způsob, jakým zařízení nakládají s přijatými zprávami. To závisí na výrobci zařízení a jeho firmware.

Mimo to je součástí standardu BACnet® také definice mnoha typů objektů a jejich vlastností, které mohou být využity interně v zařízeních implementujících tento protokol, dále algoritmy pro práci s těmito objekty (např. algoritmus vyhodnocení časového plánu nebo chybového stavu) a služby přímo sloužící ke komunikaci mezi zařízeními.

K zařízení výrobce dodává document PICS (protocol implementation conformance statement), který obsahuje definici zařízení - jeho typ, podporované objekty, atd., a také BIBBs (BACnet® interoperability building blocks) dokument, který vyjadřuje podporu služeb v zařízení.

Android aplikace

Mobilní aplikace využívající protokol BACnet®, a připojující se do sítě jako klientské BACnet® zařízení, nabízí mnoho možností využití. Lze si například představit situaci, kdy bude tato aplikace sloužit jako ovladač pro klimatizaci v hotelovém pokoji nebo kancelářském komplexu, jednoduchý způsob jak zjistit současnou teplotu nebo vypnout světla v přednáškové místnosti. Přestože se zařízení implementující protokol BACnet® využívají především v inteligentních budovách, protokol je dost obecný na to, aby bylo možné takovou aplikaci využít i v domácím prostředí.

Na druhou stranu může vzniknout servisní verze aplikace (dále nazývaná super-user verze), kterou využije například servisní technik v kotelně komplexu, aby ověřil stav všech dostupných objektů, upravil časový plán pro vytápění v případě mimořádného provozu nebo odstávky, a v pozdější verzi aplikace třeba i nahrál nový firmware do zařízení. K tomu všemu pak stačí mít na místě WiFi router pro přístup do BACnet® sítě a mobilní zařízení s takovouto aplikací.

V této práci bude do podrobna řešena verze aplikace pro běžného uživatele, která bude umožňovat pouze základní operace s objekty a bude zobrazovat

pouze hodnoty, které mají pro běžného uživatele smysl. Běžným uživatelem je myšlena osoba bez technických znalostí (takovému člověku stačí informace o teplotě v místnosti, nepotřebuje například znát tabulku priorit pro nastavení její požadované hodnoty). Aplikace bude pracovně nazvána BACdroid, tento název bude dále používán v průběhu této práce.

Kapitola 1

Analýza

1.1 BACnet®

Účelem standardu BACnet® je definice komunikačních služeb a protokolů pro počítačové vybavení použité pro monitoring a ovládání HVAC (Heating, Ventilation and Air Conditioning) a dalších systémů v budovách. Dále definuje abstraktní, objektově orientovanou reprezentaci předávání informací mezi těmito zařízeními, čímž umožňuje využití digitální ovládací technologii v budovách [1].

1.1.1 Objekty

Zařízení implementující protokol BACnet® jsou z většiny zastoupeny regulátory. Jedná se o inteligentní řídicí jednotky, na které jsou napojeny periferní zařízení, jako například čidlo teploty, servo motor ovládající klapku ventilace, nebo spínač. Tyto periferie jsou uvnitř zařízení reprezentovány objekty. Další softwarové prvky, jako časové plány a interní hodnoty, jsou také reprezentovány objekty.

Každý z objektů má sadu vlastností, ke kterým se dá přistupovat z vnějšku pomocí služeb protokolu BACnet®. Tyto vlastnosti mohou být volitelné, povinné, případně povinné a zároveň zapisovatelné. Podle těchto vlastností (a vnitřních aplikací, které přesahují rámec této práce, a ani nejsou významné pro implementovanou aplikaci) pak regulátor fyzicky ovládá hardwarové komponenty systému. Tyto vlastnosti jsou hlavním bodem zájmu této aplikace.

Součástí standardu BACnet® je sada předdefinovaných objektů. Pro každý z těchto objektů je ve standardu obsažena tabulka vlastností, která obsahuje identifikátor vlastnosti, datový typ vlastnosti, a volitelnost vlastnosti. Pro ukázkové objekty budou zmíněny vlastnosti, které mohou být relevantní pro vyvíjenou aplikaci. Některé vlastnosti jsou společné pro většinu objektů:

- `Object_Identifier` reprezentuje unikátní klíč objektu v rámci zařízení (a ve speciálních případech v celé interní síti BACnet®). Je složen z typu

zařízení (`BACnetObjectType`), který je určen výčtem, a číslem instance. Číslo instance je reprezentováno 22 bity a nesmí být rovno 4194303. Tato hodnota se využívá při referenci objektu k označení, že je tato vlastnost nedefinovaná. Jedno zařízení tedy může obsahovat až $2^{22} - 1$ objektů jednoho typu.

- `Object_Name` je pojmenování objektu, složené z jednoho a více tisknutelných znaků. Toto pojmenování musí být také unikátní v rámci zařízení.
- `Object_Type` reprezentuje typ objektu zmíněný výše.
- `Property_List` obsahuje pole (`BACnetARRAY`) identifikátorů vlastností (`BACnetPropertyIdentifier`), které jsou v daném objektu obsaženy. Slouží k ověření přítomnosti volitelných vlastností.
- `Description` je volitelnou vlastností reprezentující textový popis objektu.
- `Priority_Array` reprezentuje pole prioritních příkazů k nastavení hodnoty vlastnosti udávající současnou hodnotu objektu - `Present_Value`. Vlastnost je v objektu přítomna pouze pokud je `Present_Value` zapisovatelná, a je určena pouze ke čtení, zápis je prováděn interně. Tento mechanismus je dále popsán v kapitole Priorita příkazů.
- `Relinquish_Default` představuje výchozí hodnotu pro `Present_Value`, která je použita v případě, že je pole `Priority_Array` prázdné. Vlastnost je taktéž přítomna pouze v případě, že je `Present_Value` zapisovatelná.

Pole (`BACnetARRAY`) jsou v protokolu BACnet® základní strukturou reprezentující kolekci více prvků stejného datového typu. Jednotlivé prvky jsou zpřístupněny pomocí indexu, který je typu nezáporného celého čísla. Na pozici indexu 0 je umístěn speciální prvek, který udává počet prvků pole. Běžné prvky se tedy indexují podle základu 1. Pole je možné přečíst celé, pokud je při čtení vypuštěn parametr indexu. Pokud je možné, že se zápisem do pole změní jeho velikost, musí být prvek na pozici 0 zapisovatelný. V případě snížení hodnoty prvku na pozici 0 dojde ke zkrácení pole a prvky s indexem vyšším než nová hodnota na pozici 0 jsou smazány. Pokud je hodnota na pozici 0 zvýšena, jsou vytvořeny nové prvky. Jejich inicializace je interní záležitostí. Pokud lze měnit velikost pole, dojde při zapsání celého nového pole (zápis bez indexace) ke změně velikosti pole. Standard definuje v určitých případech pole se zadanou velikostí. Pro tato pole nelze změnit jejich velikost.

Seznamy (`BACnetLIST`) reprezentují sekvenci nula a více prvků stejného datového typu. Délka seznamu může být proměnná, a pokud není specifikováno jinak, není nijak omezena. Seznam se liší od pole tím, že k jeho prvkům nelze

přistupovat přímo pomocí indexu. Je nutné použít poziční čtení pomocí služby `ReadRange`. Navíc nelze přímo určit počet prvků v seznamu. Je třeba načíst všechny prvky a provést výpočet jejich počtu.

1.1.1.1 Device

Objekt `Device` (zařízení) reprezentuje externě dostupné charakteristiky BACnet® zařízení. V každém zařízení implementující protokol BACnet® je tento objekt obsažen právě jednou. Vlastnost `Object_Identifier` je v tomto případně unikátní nejen v rámci zařízení, ale také v rámci interní sítě BACnet®, a slouží k jednoznačné identifikaci zařízení.

Vlastnosti

- `Object_List` reprezentuje pole objektů obsažených v zařízení. Tato vlastnost může sloužit zejména v případné super-user verzi aplikace, k vyhledání všech objektů v okamžiku autodiscovery, tedy pokud např. technik přijde do oblasti, pro kterou ještě nemá vytvořen konfigurační soubor.
- `System_Status` obsahuje jeden z možných stavů, jejichž přesné chování není v standardu definováno. Označují ale fyzický a logický stav zařízení co se týče čitelnosti objektů, správy firmwaru a zálohy stavu objektů a aplikací zařízení.
- `Location` je volitelnou vlastností zařízení a slouží k popisu fyzického umístění zařízení v budově.

1.1.1.2 Analogové objekty

Tyto objekty jsou soustředěny na hodnoty reprezenotované jako číslo s pohyblivou desetinnou čárkou, konkrétně o velikosti 32 bitů.

Vlastnosti společné pro analogové objekty

- `Present_Value` reprezentuje současnou hodnotu objektu, například naměřenou teplotu.
- `Units` reprezentuje fyzikální jednotky, v kterých je udávána hodnota `Present_Value`.
- `Min_Pres_Value` definuje minimální hodnotu, které může nabývat vlastnost `Present_Value`, aby byla stále spolehlivě zjistitelná.
- `Max_Pres_Value` definuje maximální hodnotu, které může nabývat vlastnost `Present_Value`, aby byla stále spolehlivě zjistitelná.

Analog Input Analogový vstup reprezentuje informace o fyzickém vstupním zařízení. Může jít například o reprezentaci teploměru, tlakoměru a podobných zařízení. V případě, že je vlastnost `Out_Of_Service` objektu **Analog Input** nastavena na `TRUE`, musí být vlastnost `Present_Value` zapisovatelná.

Analog Output Analogový výstup reprezentuje fyzické výstupní zařízení. Slouží například k povelování klapek a serv, které mají analogovou povahu. `Present_Value` je v případě tohoto objektu vždy zapisovatelná, tím pádem jsou přítomné vlastnosti `Priority_Array` a `Relinquish_Default` pro práci s hodnotou.

Analog Value Analogová hodnota reprezentuje interní proměnnou regulátoru, může být využita například k nastavení požadované teploty v místnosti, nebo k interním výpočtům algoritmu. `Present_Value` může, ale nemusí být povelovatelná.

1.1.1.3 Binární objekty

Binární objekty jsou soustředěny na hodnoty reprezentované jako jedna z dvou možností, v případě BACnet® protokolu jsou stavy nazvány `ACTIVE` a `INACTIVE`.

Vlastnosti společné pro binární objekty

- **`Present_Value`** reprezentuje současnou hodnotu objektu, například zda je klimatizace v provozu.
- **`Polarity`** určuje, zda logický stav `INACTIVE`, resp. `ACTIVE` udává, zda je hodnota pro vstupní/výstupní zařízení převrácena nebo ne. Pokud je Polarity nastavena na `REVERSE`, stav `ACTIVE` je pro fyzické zařízení přeložen na `INACTIVE` a stav `INACTIVE` je přeložen na `ACTIVE`.
- **`Inactive_Text`** reprezentuje zobrazitelný text pro stav `INACTIVE`. Tato vlastnost je volitelná.
- **`Active_Text`** reprezentuje zobrazitelný text pro stav `ACTIVE`. Tato vlastnost je volitelná.

Binary Input Binární vstup reprezentuje informace o fyzickém zařízení nebo hardwarovém vstupu. Jedná se například o informace, zda je ventilátor nebo čerpadlo v provozu. V případě, že je vlastnost `Out_Of_Service` objektu **Binary Input** nastavena na `TRUE`, musí být vlastnost `Present_Value` zapisovatelná.

Binary Output Binární výstup reprezentuje fyzické výstupní zařízení, nebo hardwarový výstup. Typickým využitím je přepínání určitého zařízení, například světel, mezi stavy vypnuto a zapnuto. `Present_Value` je v případě tohoto objektu vždy zapisovatelná, tím pádem jsou přítomné vlastnosti `Priority_Array` a `Relinquish_Default` pro práci s hodnotou.

Binary Value Binární hodnota reprezentuje interní ovládací parametr systému, vyskytuje se v paměti BACnet® zařízení. `Present_Value` může, ale nemusí být povelovatelná.

1.1.1.4 Vícestavové objekty

Vícestavové objekty reprezentují jeden z mnoha stavů, které mohou různé hardwarové prvky nabývat. Stav je určen hodnotou typu `Unsigned`, tedy celého kladného čísla nebo nuly. V případě vícestavové hodnoty však nesmí nabýt hodnoty 0, z důvodu indexace polí - viz 1.1.1. Způsob, kterým je tato hodnota interpretována do hardwarového prvku, je interní záležitost zařízení a pro účely aplikace není třeba tento mechanismus znát.

Vlastnosti společné pro vícestavové objekty

- `Present_Value` reprezentuje současnou hodnotu objektu.
- `Number_Of_States` udává počet stavů, kterých hodnota `Present_Value` může nabýt.
- `State_Text` je pole udávající zobrazitelné texty popisující jednotlivé stavy. Pole má velikost shodnou s hodnotou vlastnosti `Number_Of_States`. Tato vlastnost je volitelná.
- `Alarm_Values` je seznam stavů, které jsou označeny jako poplachové. Tyto stavy jsou využity v algoritmu událostí. Tato vlastnost je volitelná.
- `Fault_Values` je seznam stavů, které jsou označeny jako chybové. Tyto stavy jsou využity v algoritmu chyb. Tato vlastnost je volitelná.

Multi-state Input Vícestavový vstup reprezentuje výsledek algoritmického procesu uvnitř BACnet® zařízení. Jedná se například o kombinaci několika binárních vstupů, nebo výsledek matematického výpočtu. V případě, že je vlastnost `Out_Of_Service` objektu `Multi-state Input` nastavena na `TRUE`, musí být vlastnost `Present_Value` zapisovatelná.

`Multi-state Output` Vícestavový výstup reprezentuje požadovaný stav jednoho nebo více fyzických výstupů, nebo hardwarový výstup. Typickým případem je aktivní nebo neaktivní stav několika fyzických výstupů, nebo hodnota analogového výstupu. Může sloužit například k nastavení rychlosti ventilátoru. `Present_Value` je v případě tohoto objektu vždy zapisovatelná, tím pádem jsou přítomné vlastnosti `Priority_Array` a `Relinquish_Default` pro práci s hodnotou.

`Multi-state Value` Vícestavová hodnota reprezentuje interní ovládací parametr systému, vyskytuje se v paměti BACnet® zařízení. Může například reprezentovat aktivní nebo neaktivní stav několika fyzických vstupů nebo výstupů zařízení. `Present_Value` může, ale nemusí být povelovatelná.

1.1.1.5 Schedule

Objekt `Schedule` (časový plán) je využitý k popisu periodického časového plánu, který se může opakovat v průběhu různé škály dat, s možnými výjimkami v libovolných časech a dnech. Slouží také k spojení těchto naplánovaných časů se zápisem určitých hodnot do určených vlastností jiných objektů.

Časové plány jsou rozděleny do dní, pro které jsou uvedeny dva typy:

- běžné dny během týdne,
- výjimečné dny.

Oba typy záznamů mohou definovat události pro celé trvání dne nebo jeho části. K určení, která naplánovaná událost je aktivní v určitém čase (výjimečné a běžné události se mohou překrývat), je použit prioritní algoritmus.

Vlastnosti

- **`Present_Value`** udává současnou hodnotu události časového plánu, která je určená prioritním algoritmem.
- **`Weekly_Schedule`** je pole sedmi položek, udávajících plány jednotlivých dnů v týdnu. Vlastnost je volitelná, ale pokud není přítomna vlastnost `Exception_Schedule`, musí být obsažena (a naopak).
- **`Exception_Schedule`** je pole libovolné velikosti, udávající výjimečné události oproti událostem v poli `Weekly_Schedule`. Časový údaj může být uveden jako kalendářová událost, nebo přímo reference na objekt `Calendar`. Událost dále obsahuje údaj o své prioritě.
- **`Effective_Period`** udává rozsah dat, mezi kterými je tento časový plán aktivní.

- `Schedule_Default` udává hodnotu, která je zapsána do `Present_Value` v případě, že není aktivní žádná jiná událost časového plánu.
- `List_Of_Object_Property_References` je seznam objektů, pro které je nastavována `Present_Value` do udané vlastnosti v okamžiku změny události. Tyto objekty mohou být umístěny ve stejném zařízení jako objekt časového plánu, nebo v jiném BACnet® zařízení ve stejné BACnet® interní síti. V případě, že se zařízení nepodaří z nějakého důvodu zapsat hodnotu do jednoho z objektů, proces zapisování se nezastaví a dochází k pokusu o zapsání do ostatních objektů.
- `Priority_For_Writing` udává prioritu, pod kterou bude zapisována hodnota `Present_Value` do referencovaných objektů a jejich vlastností.

1.1.1.6 Trend Log

Objekt `Trend Log` (záznam hodnot) monitoruje vlastnost referencovaného objektu, a pokud dojde ke splnění podmínek, uloží tuto hodnotu a časovou značku do interního bufferu. K záznamu může docházet periodicky, při změně hodnoty, nebo zápisem do vlastnosti `Trigger`. Hlídaný objekt může být ve stejném zařízení jako objekt `Trend Log`, nebo v jiném zařízení dosažitelném prostřednictvím interní sítě BACnet®.

Hodnota je zaznamenávána jako datový typ `BACnetLogRecord`. Může obsahovat hodnotu referencované vlastnosti, nebo změnu stavu objektu `Trend Log`. V případě problému při čtení referencované hodnoty je uložen záznam o chybě.

Záznam je uchováván v interním bufferu, který má optimálně fixní velikost. Postupně dochází k jeho zaplnění hodnotami, a pokud je plný, je nejstarší hodnota přepsána hodnotou novou, nebo dojde k zastavení monitorování referencovaného objektu. Ke čtení hodnot bufferu je využita služba `ReadRange` (více v sekci 1.1.2).

Vlastnosti

- `Enable` je hodnota typu `BOOLEAN`, která ovládá, zda dochází k zaznamenávání hodnot (`TRUE`) nebo ne (`FALSE`). K zaznamenávání musí být splněny také časové podmínky.
- `Start_Time` udává datum a čas, od kterého má docházet k zaznamenávání hodnot. Pokud je hodnota vlastnosti nedefinovaná, nebude brána v potaz. Pokud je hodnota časově po hodnotě `Stop_Time`, k zaznamenávání nedochází. Tato vlastnost je volitelná, a pokud je v objektu přítomná, je zapisovatelná.
- `Stop_Time` udává datum a čas, do kterého má docházet k zaznamenávání hodnot. Pokud je hodnota vlastnosti nedefinovaná, nebude brána

v potaz. Tato vlastnost je volitelná, a pokud je v objektu přítomná, je zapisovatelná.

- `Log_DeviceObjectProperty` je referencí na určitou vlastnost určitého objektu ve stejném zařízení jako objekt `Trend Log`, nebo v jiném BACnet® zařízení. Pokud je tato vlastnost zapisovatelná, smí být omezena pouze k referencování objektů ve stejném zařízení.
- `Buffer_Size` udává maximální počet záznamů, které může buffer udržovat.
- `Log_Buffer` je seznamem záznamů hodnot a časových značek, maximálně velikosti `Buffer_Size`. Jde o externě přístupnou reprezentaci bufferu.
- `Record_Count` udává současný počet záznamů uložených v bufferu. Zapsáním hodnoty 0 dochází k vymazání všech záznamů v bufferu.
- `Logging_Type` udává způsob určování, kdy má dojít k zaznamenání hodnoty do bufferu. Může nabývat hodnot `COV` (zaznamenání při změně hodnoty), `POLLED` (periodické zaznamenání) nebo `TRIGGERED` (na vyžádání).
- `Trigger` slouží k vyžádání zapsání hodnoty do bufferu, což se provádí zapsáním hodnoty `TRUE`. Po zaznamenání hodnoty do bufferu, nebo v případě chyby, je hodnota nastavena zpět na `FALSE`. Hodnota může být zapsána jak síťově viditelným příkazem (služba `WriteProperty` - viz 1.1.2) tak interně.

1.1.1.7 Další objekty

BACnet® podporuje mnoho dalších typů objektů, které by mohly být využity v dalších verzích aplikace. Zde jsou uvedeny nejzajímavější z těchto objektů:

- `Accumulator` slouží k periodickému načítání hodnoty, jako třeba spotřeby energie, vody, nebo plynu. Objekt umožňuje předškálovat příchozí signál pomocí násobení nebo dělení modulo a následně škálovat výslednou hodnotu pro zobrazení v určených jednotkách.
- `Calendar` reprezentuje seznam kalendářních dat. Lze o nich uvažovat jako o svátcích, speciálních událostech, nebo prostém seznamu dat. V seznamu mohou být začleněna prostá data, rozsahy dat, nebo specifikace měsíce, týdne nebo dne v týdnu (`BACnetWeekAndDay`). Pokud současné datum odpovídá některému ze záznamů, hodnota `Present_Value` je nastavena na `TRUE`, v opačném případě na `FALSE`.
- `File` obsahuje informace o souboru, který může být přístupný pomocí souborových služeb (více v sekci 1.1.2).

1.1.2 Služby

K přístupu k zařízením a jejich objektům se používají služby. Jde v podstatě o zprávy zasílané po BACnet® síti, ať už jde o broadcast nebo unicast. Zprávy zasílané službami mohou být potvrzované nebo nepotvrzované. Potvrzení může být jen prostou odpovědí, že byla zpráva přijata, složenou odpovědí (například při čtení hodnoty vlastnost je odpovědí daná hodnota) nebo chybové potvrzení, určující typ a kód chyby, kterou služba způsobila, nebo která nastala při jejím zpracování.

1.1.2.1 Služby správy vzdálených zařízení

Skupina služeb pro správu vzdálených BACnet® zařízení umožňuje zjišťovat přítomnost určitých zařízení a ovládání jejich stavů. Pomocí služby je například možné vzdáleně restartovat zařízení nebo ho na dočasnou dobu odpojit od vnější komunikace. Zde jsou uvedeny služby, které jsou zejména podstatné pro účel aplikace.

`Who-Is` je nepotvrzovanou službou, která má za účel nalezení zařízení přítomných v BACnet® interní síti. Může být také využita k zjištění adresy zařízení, pro které je známo identifikační číslo. Služba má dva nepovinné parametry:

- **`Device Instance Range Low Limit`** určije spodní hranici identifikačního čísla zařízení, které má na identifikační požadavek odpovídat. Nejnižší možnou hodnotou je 0.

- **`Device Instance Range High Limit`** určije horní hranici identifikačního čísla zařízení, které má na identifikační požadavek odpovídat. Nejvyšší hodnotou parametru může být 4194303.

Pokud jsou parametry vynechány, jakékoliv zařízení v BACnet® síti může odpovědět zasláním služby `I-Am`. Pro získání adresy zařízení se známým ID se použije služba `Who-Is` s oběma parametry nastavenými právě na hodnotu tohoto identifikačního čísla. Stejnou službu lze použít k ověření, zda je zařízení se zadaným identifikačním číslem přítomno v síti.

`I-Am` je nepotvrzovanou službou určenou k odpovědi na službu `Who-Is`. Může být zaslána jako broadcast, nebo konkrétnímu příjemci. Součástí odpovědi je identifikace zařízení, maximální velikost zprávy, kterou je zařízení schopno přijmout, informace, zda zařízení podporuje segmentaci zpráv a identifikaci výrobce zařízení.

Who-Has je nepotvrzovanou službou určenou ke zjištění identifikačních čísel a adres zařízení, které obsahují objekty se zadaným názvem nebo identifikátorem (kombinace identifikačního čísla a typu objektu). Podobně, jako služba Who-Is, i tato služba umožňuje jako parametry zadat horní a dolní hranici identifikátoru zařízení. Alespoň jeden z parametrů název nebo identifikátor objektu musí být zadán.

I-Have je nepotvrzovanou službou určenou k odpovědi na službu Who-Has. Služba je vždy zasílána jako broadcast. Součástí zprávy je identifikátor zařízení, identifikátor objektu a název objektu. Aby mohlo zařízení odpovědět, musí jeho identifikátor spadat do rozmezí zadané službou Who-Has. Pokud je službou Who-Has zadán identifikátor objektu, odpovídá pouze zařízení obsahující objekt s tímto identifikátorem. Pokud je zadán službou Who-Has název objektu, odpovídá pouze zařízení, které obsahuje objekt s tímto názvem.

1.1.2.2 Služby pro přístup k objektům

Skupina služeb určených pro přístup k objektům slouží k čtení a zápisu hodnot do objektů a k přímé manipulaci s objekty. Je možné vzdáleně mazat nebo vytvářet objekty, měnit prvky seznamů nebo přistupovat k prvkům pole. Zde jsou zmíněny služby pro přístup k objektům, které jsou zejména podstatné pro fungování aplikace. Mimo standardní definované objekty a vlastnosti lze tyto služby využít i pro přístup k objektům a vlastnostem definovaným výrobcem zařízení.

ReadProperty je potvrzovaná služba určená k vyžádání hodnoty jedné vlastnosti jednoho objektu. Parametry jsou identifikátor objektu a identifikátor vlastnosti, kterou chceme číst. Pokud je čtenou vlastností pole, je možné zadat ještě parametr indexu prvku pole, který chceme číst. Odpovědí může být buď kopie těchto parametrů včetně hodnoty požadované vlastnosti, nebo chybová odpověď obsahující informaci o typu chyby. K chybě dojde, pokud:

- zadaný objekt neexistuje,
- zadaná vlastnost neexistuje,
- zadaná vlastnost není typu pole, ale byl zadán parametr indexu,
- zadaný index je mimo rozsah pole,
- zadaná vlastnost není přístupná pro čtení touto službou.

ReadPropertyMultiple je potvrzovaná služba podobná službě ReadProperty. Slouží ke čtení hodnot jedné a více vlastností jednoho a více objektů. Parametrem je seznam specifikací čtení. Specifikace čtení je zadána jako identifikátor

objektu a seznam referencí vlastností. Odpověď v případě úspěšného čtení obsahuje seznam výsledků. Výsledek je reprezentován identifikátorem objektu a seznamem vlastností, včetně jejich hodnot a případných přístupových chyb. Mohou nastat stejné chyby jako v případě služby `ReadProperty`.

`ReadRange` je potvrzovaná služba určená ke čtení hodnot ze seznamů a polí seznamů. Parametry služby jsou identifikátor objektu a identifikátor vlastnosti. V případě, že je vlastnost typu pole seznamů, je zadán parametr indexu prvku pole, ze kterého se má číst. Dalším parametrem je rozsah čtení, který může být udán několika způsoby:

- **Podle pozice** je zadána pozice prvního (pokud je počet čtených prvků kladný) nebo posledního (pokud je počet čtených prvků záporný) prvku, který má být čten, a dále počet prvků k čtení. Pozice je brána vzhledem k začátku seznamu.
- **Podle sekvenčního čísla** je zadáno sekvenční číslo prvku, od/do kterého se má číst (v závislosti na znaménku počtu prvků). Sekvenční číslo je prvku přiřazeno v okamžiku přidání prvku do seznamu. Číslo pozice se odebíráním prvků může měnit, sekvenční číslo nikoliv.
- **Podle času** je zadán údaj časové známky, od/do které se mají prvky seznamu číst (v závislosti na znaménku počtu prvků). Je čteno od prvního prvku s novější časovou známkou, než je zadána, nebo do prvku s poslední časovou známkou která je starší než zadaná časová známka.

`WriteProperty` je potvrzovaná služba určená k zápisu hodnoty jedné vlastnosti jednoho objektu. Parametry jsou identifikátor objektu a identifikátor vlastnosti, do které se má zapsat. Pokud je referencovaná vlastnost typu pole, je přidán parametr indexu prvku pole, do kterého se má zapsat. Pokud je v tomto případě tento parametr vynechán, bude přepsáno celé pole. Dalšími parametry jsou hodnota k zapsání do vlastnosti a priorita, podle které se má do vlastnosti zapsat (viz 1.1.3). V případě úspěchu je odpovědí jednoduché potvrzení, v případě neúspěchu je udán typ chyby, která nastala. Mimo chyb zmíněných u služby `ReadProperty` může dojít k následujícím typům chyb:

- zadaná vlastnost není přístupná pro zápis,
- datový typ zadané hodnoty není kompatibilní s vlastností pro zápis,
- vlastnost je typu `Object_Name` a zadaná hodnota názvu objektu už se v zařízení vyskytuje,
- vlastnost je typu `Object_Identifier` a zadaná hodnota identifikátoru už se v zařízení vyskytuje,

- zadaná hodnota je mimo povolený interval hodnot, které zadaná vlastnost podporuje,
- není dostatek místa k zapsání hodnoty,
- parametr priority zápisu je mimo rozsah (1 - 16).

`WritePropertyMultiple` je potvrzovaná služba, která je velmi podobná službě `WriteProperty`. Slouží k zápisu hodnot do jedné nebo více vlastností jednoho nebo více objektů. Parametrem služby je seznam specifikací zápisu. Specifikace zápisu je zadána identifikátorem objektu a seznamem referencí vlastností, hodnot a priorit, kterými mají být hodnoty do vlastností zapsány. V případě neúspěchu je uveden typ chyby a identifikátor objektu, identifikátor vlastnosti a index vlastnosti, která způsobila chybu zápisu.

`AddListElement` je potvrzovaná služba určená k přidání jednoho nebo více prvků k určené vlastnosti objektu, která je typu seznam (`BACnetLIST`) nebo pole seznamů. Parametry služby jsou identifikátor objektu a identifikátor vlastnosti. Pokud je referencovaná vlastnost typu pole seznamů, je přidán volitelný parametr index prvku pole. Data k zápisu jsou předána rovněž jako seznam. V případě úspěchu je služba potvrzena jednoduchou odpovědí. V případě neúspěchu je uveden typ chyby a v případě, že je chyba způsobena parametrem seznamu prvků k zapsání, je uveden index prvku, který jako první způsobil chybu. Tento index je počítaný od 1, v případě indexu 0 není původce chyby v seznamu prvků k zapsání.

`RemoveListElement` je potvrzovaná služba určená k odebrání jednoho nebo více prvků z určené vlastnosti objektu, která je typu seznam nebo pole seznamů. Tato služba nepodporuje vnořené seznamy. Pokud je prvkem určeným k odebrání vnořený seznam, je odebrán celý. Parametry jsou identické s parametry služby `AddListElement`.

1.1.2.3 Služby pro přístup k souborům

Tato skupina služeb umožňuje přístup k souborům umístěným v BACnet® zařízeních. Koncept souborů je v tomto případě abstraktní a neimplikuje přítomnost disku nebo jiného paměťového média pro uložení souborů v tradičním slova smyslu. Každý soubor, který je dostupný, bude reprezentován odpovídajícím objektem typu `File`. K přístupu jsou používány služby `AtomicReadFile` a `AtomicWriteFile`. „Atomické“ v tomto případě značí, že v okamžiku čtení nebo zápisu není možné k souboru přistupovat jiným voláním služby `AtomicReadFile` nebo `AtomicWriteFile`. Způsob synchronizace těchto operací s interním přístupem k souborům je závislý na výrobci.

K souborům je možné přistupovat proudově, tedy po jednotlivých bytech, nebo po záznamech.

1.1.2.4 Další služby

Protokol BACnet® definuje mnoho dalších služeb, které přesahují rámec této práce. Jejich využití v pozdějších verzích aplikace ovšem není vyloučeno. Jde hlavně o služby, které umožňují přihlášení k odběru určitých událostí od zařízení.

- **COV (change of value - změna hodnoty)** je skupinou služeb, které umožňují odebírat upozornění o změně hodnoty vlastnosti objektu. Přihlášením k odběru je odběratel zaznamenán do seznamu odběratelů v objektu Device. V okamžiku změny hodnoty o požadovaný přírůstek je seznamu odběratelů odeslána informace o změně hodnoty, potvrzovanou nebo nepotvrzovanou službou.
- **Poplachy a události** je skupina služeb podobná skupině COV, zařízení se ale hlásí k odběru poplachových a chybových událostí.

1.1.3 Priorita příkazů

V systémech ovládání budov mohou být objekty manipulovány mnoha entitami. K objektům mohou přistupovat aplikace obsažené v BACnet® zařízeních, běžní uživatelé a správci budovy. Z toho důvodu je v protokolu BACnet® zaveden mechanismus priority příkazů. V tomto mechanismu figurují tyto vlastnosti objektů:

- `Commandable Property` **(povelovatelná vlastnost):** Každý objekt podporující prioritizaci příkazů má jednu nebo více vlastností, které jsou označeny jako povelovatelné. Hodnota této vlastnosti je ovládána mechanismem prioritizace.
- `Priority_Array` **(pole priorit):** Tato vlastnost objektu, určená pouze ke čtení, obsahuje prioritizované příkazy nebo hodnoty `NULL`, seřazené podle klesající priority.
- `Relinquish_Default` **(výchozí hodnota uvolnění):** Tato vlastnost je stejného datového typu jako povelovatelná vlastnost. Když mají všechny položky pole priorit hodnotu null, povelovatelná vlastnost bude mít hodnotu této vlastnosti.

Velikost pole priorit je pro BACnet® objekty fixní. Priorita příkazu může nabývat hodnoty mezi 1 a 16, kde 1 je nejvyšší priorita, 16 nejnižší. Prioritní povel se provádí pomocí služby `WriteProperty` nebo `WritePropertyMultiple` (viz 1.1.2.2). Priorita je volitelným parametrem těchto služeb. Pokud je vynechána a je prováděn zápis do povelovatelné vlastnosti, je příkazu přiřazena výchozí priorita 16. V případě, že je priorita udána pro zápis do vlastnosti, která není pod správou prioritního mechanismu, je tato hodnota zanedbána.

Pole priorit má přiřazené některé výchozí priority. Ostatní priority jsou volně přiřaditelné aplikacím a závisí na konkrétním použití zařízení, jak budou tyto priority přiděleny. Tabulka 1.1 obsahuje standardní rozdělení priorit.

Záznam v prioritním poli může mít hodnotu povelu, nebo hodnotu `NULL`. Hodnota `NULL` značí, že pro danou prioritu není zadán žádný povel. Objekt průběžně monitoruje pole priorit aby určil hodnotu povelované vlastnosti. Povelující entita (aplikace, technik, uživatel, atd.) může zadat požadavek pro zapsání hodnoty, nebo pro uvolnění požadavku zadaného dříve (tzv. `Relinquish`). To je provedeno zadáním příkazu pro zapsání hodnoty `NULL` pro danou prioritu. Určení hodnoty povelované vlastnosti je provedeno zjištěním záznamu pole priorit s nejvyšší prioritou, která nenese hodnotu `NULL`. Pokud jsou všechny hodnoty pole priorit `NULL`, je nastavena hodnota `Relinquish_Default`.

V případě, že je zadán požadavek na povelování s určitou prioritou, a v tabulce priority existuje předchozí povel s touto prioritou, jeho hodnota je přepsána bez obeznámení entity, která provedla předchozí zápis. Z toho důvodu je doporučeno každé aplikaci přiřadit jinou prioritu pro zápis. Stejně se mechanismus chová i při uvolnění hodnoty.

V momentě zápisu se hodnota vlastnosti v zařízení skutečně změní pouze v případě, že tabulka obsahuje v řádcích vyšší priority pouze prázdné hodnoty. Pokud obsahuje řádek s vyšší prioritou hodnotu jinou než `NULL`, zůstane hodnota vlastnosti nezměněná. Krom zápisu lze také uvolnit hodnotu pro příslušnou prioritu. Pokud tato hodnota korespondovala s nejvyšší zapsanou

Tabulka 1.1: Standardní priority příkazů

Priority Level	**Application**
1	Manual-Life Safety
2	Automatic-Life Safety
3	Available
4	Available
5	Critical Equipment Control
6	Minimum On/Off
7	Available
8	Manual Operator
9	Available
10	Available
11	Available
12	Available
13	Available
14	Available
15	Available
16	Available

prioritou v tabulce, změní se hodnota vlastnosti na hodnotu zapsanou v nejbližší nižší prioritě. Pokud žádná taková není, je hodnota nastavena na výchozí hodnotu, která je uvedená v hodnotě `Relinquish_Default`.

1.2 Obdobné aplikace

Pro operační systém Android již existuje několik aplikací, které komunikují s regulátory pomocí protokolu BACnet®. Všechny tyto aplikace jsou placené, nebylo tedy možné je přímo otestovat a k jejich porovnání tedy sloužily pouze informace dostupné na stránkách obchodu Google Play™, případně webových stránkách aplikací.

Při porovnávání byly brány v potaz následující kritéria:

1. podporované objekty,
2. přehlednost uživatelského rozhraní,
3. uživatelská přívětivost,
4. způsob vytváření seznamu objektů k zobrazení.

1.2.1 BACnet Explorer

BACnet Explorer - Google Play

1. Aplikace podporuje čtení a zápis pouze pro základní objekty - analogové, binární a vícestavové objekty. Umožňuje číst a povelovat jejich hodnoty. Je schopna číst hodnoty dostupných vlastností BACnet® objektů. Navíc umožňuje základní práci se soubory a jejich ukládání do paměti Android zařízení.
2. Uživatelské rozhraní je prosté a přímočaré. Neklade důraz na zobrazení jednotlivých hodnot, ale spíše na přehledy objektů, aplikace je cílená na technicky zdatné uživatele.
3. Aplikace slouží spíše pro technicky znalé uživatele, kteří jsou obeznámeni s komunikačním protokolem BACnet®. Pro běžného uživatele nebo domácnost tato aplikace není vhodná, ale cílovou skupinou jsou spíše servisní technici.
4. Objekty jsou mapovány funkcí autodiscovery, pomocí služeb `Who-Is` a `I-Am`. Dále umožňuje manuální slučování objektů do seznamů.

Tato aplikace se blíží super-user verzi, která bude dále rozepsána v kapitole Další rozvoj. Funkce vytváření seznamů zařízení je užitečná a stálo by za uvážení, zda ji začlenit do aplikace BACdroid. Vzhledem k cílové skupině ale bude spíše sloužit jako inspirace pro budoucí super-user verzi.

1.2.2 BACnet HMI

BACnet HMI - Google Play

1. Aplikace podporuje stejné objekty jako předchozí aplikace.
2. Uživatelské rozhraní je elegantní a přívětivé. Důraz je kladen na zobrazení hodnot důležitých vlastností objektů, které jsou spojované do logických skupin. Působivým prvkem je zobrazení grafů, není ovšem zřejmé, zda jde o zobrazení objektu `Trend Log`, či je záznam hodnot prováděn interně.
3. Aplikace je cílená na běžného uživatele, zobrazuje hodnoty tak, jak by je měl být schopen vyčíst i člověk, který se nikdy nesetkal s průmyslovým regulátorem nebo jakýmkoliv technickým prvkem HVAC.
4. Konfigurace je prováděna pomocí JSON souborů, které obsahují velké množství informací o zařízení a napojení na některé grafické prvky aplikace.

Aplikace se mnohem více blíží představě o aplikaci BACdroid (název HMI značí human-machine interface, tedy rozhraní mezi strojem a člověkem). Je dílem stejné firmy jako aplikace BACnet Explorer, ale zdá se, že není konečným řešením pro stranu systému Android. Podle dokumentace je nástrojem k vytváření aplikací pomocí webových technologií HTML5, CSS3 a JavaScript. Je tedy frameworkem pro komunikaci se zařízeními implementujícími protokol BACnet®. Hlavním přínosem aplikace je elegantní uživatelské rozhraní.

1.2.3 NetLink

NetLink - Google Play
NetLink - BACnet plugin

1. Aplikace podporuje stejné objekty jako předchozí aplikace, tedy pouze analogové, binární a vícestavové objekty.
2. Uživatelské rozhraní je velmi strohé, umožňuje však velkou řadu funkcí pro přesné zacházení s obejkty. Sdružování objektů do skupin podle typu je užitečné zejména pro případ, kdy zařízení obsahuje velké množství objektů.
3. Aplikace je cílená na technicky zdatné uživatele, umožňuje velkou úroveň řízení všech funkcí. Velice se blíží aplikaci BACnet Explorer, navíc ale zobrazuje grafy. Není zřejmé, zda jde o interně zaznamenané hodnoty, či grafickou reprezentaci objektu `Trend Log`.
4. Konfigurace je prováděna funkcí autodiscovery, tedy pomocí služeb `Who-Is` a `I-Am` a objekty jsou přidávány do sdíleného seznamu.

Aplikace se velmi blíží funkcionalitě první zmíněné aplikace - BACnet Explorer. Je tedy vhodná spíše pro servisní techniky než pro běžné uživatele. Rozhraní je strohé a neblíží se modernímu designu. Plugin pro protokol BACnet® ale umožňuje napojení na jiné aplikace, než pouze NetLink.

Tyto aplikace mají své kladné stránky, ale jsou směřovány spíše na servisní techniky než koncové uživatele. Tomu odpovídá jak uživatelské prostředí, tak možnosti ovládání jednotlivých prvků. Pro běžného uživatele není nutné poskytnout ovládání všech atributů manuálně, u každého zařízení většinou stačí ovládat jednu hodnotu, a model aplikace pak zařídí vše potřebné. Dalším velkým rozdílem od zamýšlené aplikace BACdroid je způsob konfigurace aplikace, která nemá uživateli poskytnout přístup ke všem zařízením v síti, ale pouze k těm, která jsou pro uživatele relevantní. Tomu se blíží aplikace BACnet HMI, ale pro běžného uživatele je třeba tento systém značně vylepšit. K tomu by mohly sloužit například NFC tagy nebo QR kódy, rozmístěné v jednotlivých místnostech. Ty by obsahovaly identifikaci přístrojů, které se v místnosti nacházejí, a uživatel by tak nemusel listovat v seznamu všech objektů dostupných po síti.

1.3 Existující knihovny

Vzhledem k rozsáhlosti protokolu BACnet® není možné jako součást této práce implementovat nízkoúrovňové prvky klientské části, zejména komunikační vrstvu. Implementace této knihovny by jistě přesahovala rozsah více než jedné diplomové práce. Proto je třeba vybrat z existujících knihoven takovou, která je nejvíce vhodná pro použití pro aplikaci BACdroid na platformě Android.

Kritéria pro výběr knihovny jsou následující:

1. dostupné objekty a služby,
2. složitost integrace do platformy Android,
3. licencování a úroveň podpory,
4. aktuálnost knihovny,
5. úroveň znalosti jazyka autora této práce.

1.3.1 BACnet Stack

BACnet Stack je knihovna implementovaná v jazyce C++ a poskytuje aplikační vrstvu, síťovou vrstvu a vrstvu MAC. Podporuje operační systémy MS Windows, Linux, další operační systémy i vestavěné systémy.

1. Knihovna implementuje většinu služeb, ale podpora objektů je nižší. Chybí zejména implementace objektů Schedule a Calendar, které by měly být podstatné pro pozdější verze aplikace.

2. Platforma Android umožňuje integraci knihoven psaných v jazyce C a C++ pomocí tzv. NDK (native development kit). Pro případné úpravy kódu knihovny je ale třeba využít experimentální verze Gradle - nástroje pro automatizaci a build Android projektů.

3. BACnet Stack je licencován pod upravenou verzí GPL. Veškeré úpravy knihovny musí být zveřejněny pod stejnou licencí, ale aplikace využívající tuto knihovnu na sebe licenci GPL nemusí přebírat. Dokumentace je strohá, ale součástí knihovny jsou ukázkové aplikace.

4. Vývoj knihovny stále pokračuje, poslední verze je dostupná k 4. 3. 2016.

5. Autor práce je obeznámen s jazykem C++, nemá ovšem zkušenosti s jeho integrací v platformě Android.

1.3.2 BACnet4J

BACnet4J je knihovna psaná v jazyce Java, který je standardním jazykem využívaným pro vývoj Android aplikací.

1. Knihovna implementuje velké množství služeb a díky implementovaným datovým typům umožňuje monitorování a povelování všech známých typů objektů.

2. Pro platformu Android je možná přímá integrace knihovny standardním nástrojem Gradle, bez jakýchkoliv komplikací.

3. Knihovna je licencovaná pod licencí GPL. Pro komerční využití pak tvůrce nabízí možnost za poplatek pozměnit licenci.

4. Vývoj knihovny stále pokračuje, poslední verze je dostupná od 19. 2. 2016

5. Autor práce má rozsáhlé zkušenosti s vývojem aplikací v jazyce Java a použitím tohoto jazyka při práci na projektech pro platformu Android.

1.3.3 BACpypes

BACpypes je knihovna implementovaná v jazyce Python.

1. Knihovna implementuje většinu podporovaných objektů a základní sadu služeb.

2. Integrace do platformy Android probíhá pomocí nástroje SL4A (Scripting Layer for Android), který umožňuje spouštění skriptů v několika jazycích. Napojení na uživatelské rozhraní ale není triviální. Tento projekt již není dále vyvíjen.

3. Informace o licencování nejsou dostupné a dokumentace není velice přehledná. Neobsahuje například seznam podporovaných služeb. Obsahuje ukázky kódu.

4. Vývoj knihovny pokračuje, poslední verze byla zveřejněna 13. 2. 2016.

5. Autor práce nemá žádné zkušenosti s programovacím jazykem Python.

Vzhledem k výhodám a nevýhodám popsaným výše bylo rozhodnuto, že bude využita knihovna BACnet4J. Hlavními přínosy jsou snadná integrace, zkušenosti autora práce s daným programovacím jazykem a rozsah implementace knihovny. Všechny zmíněné knihovny jsou stále ve vývoji, tento bod tedy při výběru nebyl brán v potaz.

V případě, že by byly na aplikaci kladeny nároky na rychlost, stála by za uvážení knihovna BACnet Stack. Použití nativních metod a jazyka C++ má v tomto směru výhodu oproti ostatním. Povaha systému řízení vytápění, ventilace a klimatizace ovšem není příliš závislá na rychlé odezvě. Výhoda knihovny v jazyce C++ tedy není natolik podstatná.

1.4 BACnet MS/TP

Existují BACnet® zařízení, která nejsou vybavena rozhraním Ethernet a proto nemohou implementovat protokol BACnet/IP. Taková zařízení většinou používají sériové komunikační rozhraní standardu EIA-485 a implementují protokol BACnet MS/TP (Master/Slave Token Passing).

Při této multi-masterové komunikaci si jednotlivá zařízení předávají token (peška) a ostatním zasílají data pouze v případě, že peška mají. Toho si mohou ponechat pouze na určenou dobu, aby nedocházelo k asymetrickému vytížení komunikace. Předáním peška umožní dalšímu zařízení posílat data.

Tento algoritmus vyžaduje přesné časování, které umožňuje novým zařízením zařadit se do kruhu pro předávání peška bez toho, aby byla rozbita komunikace ostatních zařízení.

Takovýto způsob komunikace není pro Android zařízení nativní, komerční zařízení navíc nejsou vybavena sériovou linkou standardu EIA-485 ani RS-232. Tento problém je možné obejít využitím USB portu a převodníku. Pro jeho fungování musí USB port zařízení podporovat USB On-The-Go, umožňující zařízení chovat se jako USB host. Mimo běžné mobilní zařízení existují i specializované vestavné systémy, které mají zabudovaný konektor sériové linky. V tom případě odpadá nutnost využít převodníku a USB portu.

Alternativní možností připojení aplikace BACdroid k zařízením typu BACnet MS/TP je použití BACnet® routeru. BACnet® routery směrují zprávy mezi sítí IP a MS/TP. Knihovna BACnet4J routování nativně podporuje.

Android jako systém s linuxovým jádrem má přístup k sériové lince mapovaný do iterních souborů ve složce /dev/. Pro práci s těmito soubory je vhodné využít knihovnu nativního kódu ke zrychlení práce, vzhledem k nízké úrovni komunikace. Taková knihovna ale může být dostupná pouze pro určitá zařízení a závislá na vnitřní struktuře. Android také od verze api 3.1 poskytuje USB Host api, které umožňuje práci s USB, kvůli nárokům na časování ovšem tento přístup nemusí být dostatečný.

Knihovna BACnet4J je připravena na práci se sériovou linkou díky abstraktní třídě *SerialPortWrapper*. Její rozšiřující implementaci je třeba napojit na knihovnu přímo pracující se sériovými porty a pak pomocí ní inicializovat síťovou vrstvu.

1.5 Požadavky na aplikaci

V rámci této diplomové práce dojde k vývoji aplikace BACdroid, která bude sloužit jako ukázka práce systému, který bude popsán níže. Tato aplikace je cílena na uživatele bez rozsáhlých technických znalostí. Uživatel nemusí vědět, co je to protokol BACnet®, jak fungují inteligentní budovy a jak probíhá jejich ovládání pomocí regulátorů.

1.5.1 Funkční požadavky

Součástí aplikace bude konfigurační systém, který umožní nahrát seznam zařízení, objektů a jejich vlastností, které jsou pro běžného uživatele podstatné, do prostředí aplikace.

Některé hodnoty této konfigurace (např. pojmenování oblastí) bude možné změnit a trvale tyto změny uložit v paměti mobilního zařízení.

Aplikace bude moci číst a povelovat dostupné objekty a jejich vlastnosti.

Aplikace bude umět reagovat na problémy způsobené komunikací s interní sítí BACnet® a jejími zařízeními. Pokud to bude možné, předá uživateli informace o povaze problému a pokusí se ho navést na způsob vyváznutí z chybového stavu.

1.5.2 Nefunkční požadavky

Konfigurace oblastí bude trvale uložena v paměti mobilního zařízení.

Aplikace bude s BACnet® zařízeními komunikovat přes lokální WiFi síť. Další možnosti komunikace nejsou vyloučeny v dalších verzích.

Aplikace bude navržena tak, aby bylo možné snadno implementovat zobrazení nových typů objektů. Součástí aplikace bude sada několika objektů,

které budou zobrazitelné a případně (pokud to BACnet® zařízení povoluje) povelovatelné.

Aplikace bude mít jednoduché, přívětivé rozhraní určené pro cílovou skupinu. Ovládání musí být intuitivní.

Uživatelské rozhraní bude využívat vodítek definovanách v Material designu [3], vytvořeným společností Google.

1.6 Případy užití

1.6.1 Načtení konfigurace

Pro první verzi aplikace bude konfigurace načtena z paměťového úložiště mobilního zařízení. Do zařízení je možné soubor dopravit pomocí emailu, USB, nebo stažením z webového serveru.

Konfiguraci by v pozdějších verzích bylo možné načítat přímo z NFC tagu na místě použití aplikace. V první verzi NFC tag slouží k vyvolání příslušné (již načtené) konfigurace podle identifikačního čísla konfigurace.

Konfigurace je vytvořena servisním technikem, který má přístup k informacím o zařízeních a objektech dostupných na místě využití aplikace. Je na uvážení správce oblasti, které objekty zpřístupní uživatelům aplikace a u kterých jim umožní objekty povelovat.

1.6.2 Seznamy objektů - oblasti

Aplikace zobrazuje objekty v seznamech tak, jak je zadáno v konfiguraci. Objekty mohou být uspořádány do adresářové struktury, aplikace pak umožní postupný průchod těmito seznamy.

Seznamy může uživatel přejmenovat a v případě celé oblasti také konfiguraci smazat. To může být užitečné zejména pokud uživatel cestuje mezi oblastmi a ví, že se už do určité oblasti nebude vracet. V konfiguraci nemusí být obsaženy lokalizované názvy, nebo může uživatel chtít využít pro pojmenování oblastí zkratky.

1.6.3 Povelování zařízení

Pokud to zařízení a objekt podporuje, a pokud je v konfiguraci vlastnost objektu označena jako povelovatelná, může uživatel zadávat honoty vlastnostem objektů. Může jít například o přepnutí objektu `Binary Output`, což má za následek vypnutí světla, nebo zadání konkrétní hodnoty do objektu `Analog Value`, kterou zařízení využívá k vyhodnocení požadované teploty v místnosti.

1.6.4 Obnovení hodnot

V živém systému se hodnoty vlastností objektů průběžně mění. Aplikace musí v průběhu svého životního cyklu udržovat aktuální hodnoty a pravidelně je

občerstvovat, pokud si to uživatel přeje.

Zároveň musí být schopna na povel uživatele obnovit hodnoty mimo pravidelný cyklus. K tomu může dojít při manuálním zadání povelu k obnovení, nebo při povelování hodnoty. Pokud uživatel manuálně mění hodnotu vlastnosti objektu, aplikace znovu načte vlastnosti objektu, aby došlo k ujištění, že byl příkaz zadán správně.

1.6.5 Seznam úkonů

Načtení konfigurace oblasti probíhá výběrem souboru v paměti telefonu. K takovému úkonu je třeba mít nainstalovanou aplikaci sloužící k prohlížení souborů. V případě, že taková aplikace nainstalovaná není, dojde k upozornění uživatele, že je aplikace vyžadována.

Načtení NFC tagu slouží k vyvolání příslušné konfigurace a zobrazení pro uživatele. Načítání NFC tagu lze provést i v případě, že je aplikace neaktivní. Systém Android předá tuto událost aplikaci. V případě, že aplikace není nainstalovaná, je uživatel přesměrován na odkaz k jejímu stažení (v případě zveřejnění aplikace je uživatel přesměrován do aplikace Obchod Google Play).

Přejmenování konfigurace oblasti je užitečné zejména v případech, kdy je v konfiguračním souboru oblast pojmenována nevhodně. Změna jména musí být trvale uložena v aplikaci.

Smazání konfigurace oblasti umožňuje změnu konfigurace, například při změně konfiguračního souboru nebo úplné změně lokace.

Zobrazení objektů v oblasti slouží jako přehled všech objektů, které jsou obsaženy v konfiguračním souboru, nebo jejich podmnožiny.

Zobrazení oblíbených objektů umožňuje uživateli mít přehled nad objekty, které využívá nejčastěji. Tyto objekty jsou viditelné ihned po spuštění aplikace.

Přidání objektu do oblíbených může probíhat pro libovolný objekt z libovolné oblasti. V případě domácnosti mohou být objekty rozděleny podle pokojů, ale uživatel používá oblíbené objekty jako dohled nad nejdůležitějšími objekty v celé domácnosti.

Odebrání objektu z oblíbených je užitečné při změně preferencí uživatele, nebo v případě, že uživatel přidá do seznamu oblíbených objektů nějaký objekt omylem.

Zobrazení detailu objektu slouží k získání informací o objektu, které nemusí být potřebné při běžném užívání aplikace. Zobrazen může být například dodatečný popis, informace o omezení hodnot, nebo grafy hodnot a časové plány.

Obnova hodnot seznamu objektů slouží k manuálnímu vyžádání nejčerstvějších hodnot ze zařízení BACnet®. Může se týkat i jednoho objektu.

Zadání hodnoty objektu - povel umožňuje uživateli změnit současnou hodnotu vlastnosti objektu. Tato akce vyvolá obnovu hodnot objektu.

Nastavení výchozí hodnoty objektu - odebrání povelu odebírá dříve zadaný povel. Tato operace je popsána v sekci Priorita příkazů, jedná se o `Relinquish` hodnoty priority, která je použitá při povelování.

Zobrazení nastavení umožňuje přejít do obrazovky nastavení, která obsahuje trvalé volby týkající se celé aplikace.

Nastavení obnovování dat umožňuje vypnout nebo zapnout pravidelné obnovování hodnot objektů.

Nastavení frekvence obnovování dat dává uživateli na výběr z různých frekvencí obnovy dat, od několika sekund až po minuty.

Zobrazení upozornění o spotřebě slouží k varování uživatele, že zadané nastavení může vést k zvýšené spotřebě baterie v případě, že k obnovování dochází příliš často. Síťové operace jsou náročné na energii.

Zobrazení informací o aplikaci v aplikaci přechází na obrazovku obsahující informaci o verzi aplikace, licenci a případné informace o práci systému nebo nápovědu.

Kapitola 2

Návrh

2.1 Model dat

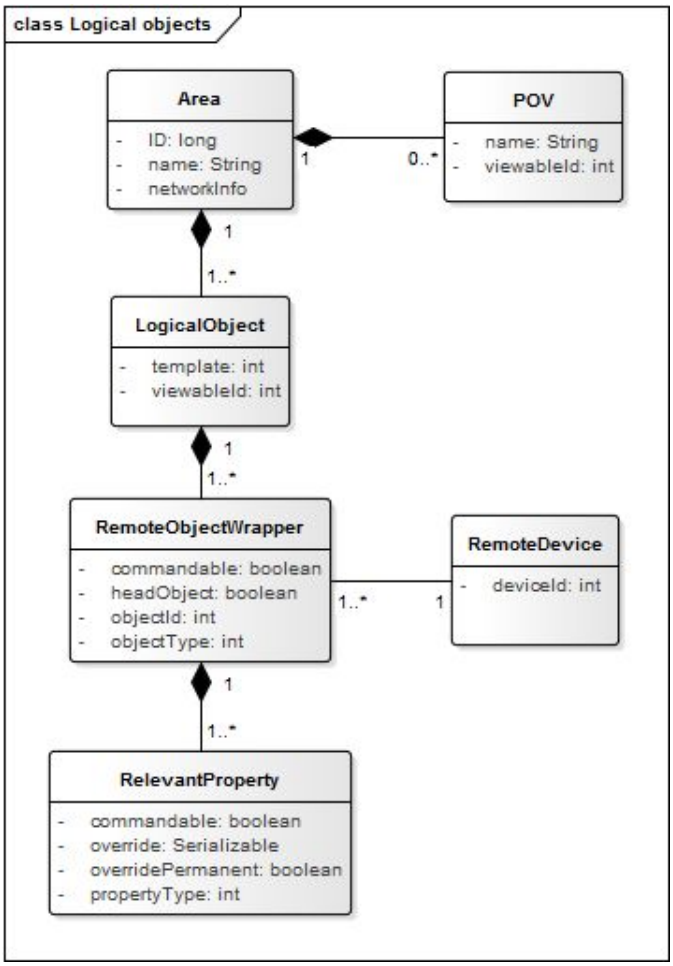
Pro účely aplikace bude zaveden model logických objektů, znázorněný na obrázku 2.1. Logické objekty budou jednotně zobrazovány v uživatelském prostředí jako celistvé prvky. Budou se skládat z jednoho nebo více BACnet® objektů. Tyto BACnet® objekty budou nazývány jako fyzické objekty, na obrázku reprezentovány třídou *RemoteObjectWrapper*, protože se vyskytují ve fyzických BACnet® zařízeních. Logické objekty budou zprostředkovávat návaznost obsažených fyzických objektů na uživatelské rozhraní a jejich vzájemné propojení. Příkladem může být složení objektů `Analog Input` (snímaná teplota), `Analog Value` (požadovaná teplota) a `Trend Log` (záznam teploty za poslední dvě hodiny). Tyto objekty budou odlišně zobrazovány v detailu a náhledu logického objektu a můžou sdílet některé vlastnosti, například jednotky (v tomto příkladě stupně celsia).

Každý fyzický objekt má referenci na příslušné BACnet® zařízení, které je v knihovně BACnet4J reprezenrováno třídou *RemoteDevice*. Vzhledem k tomu, že mnoho fyzických objektů sdílí stejné BACnet® zařízení, není nutné ho zvlášť inicializovat pro každý objekt. Zařízení je třeba objevit podle identifikačního čísla službou `Who-Is`, aby byla získána jeho síťová adresa. Poté může být třída zařízení uložena a v případě potřeby vyvolána reference bez nutnosti použití služby `Who-Is`.

Fyzický objekt dále obsahuje reference na relevantní vlastnosti, které je třeba zobrazit v různých částech rozhraní aplikace. Pro napojení hodnot na uživatelské rozhraní bude vytvořen mechanismus, který je zmíněn níže, v sekci Zobrazení objektů.

2.2 Konfigurace oblasti

Pro účely co možná největšího zjednodušení práce uživatele aplikace je třeba zavést systém konfigurace oblasti, který umožní snadno načíst seznam zařízení,



Obrázek 2.1: Model dat aplikace

jejich objektů a vlastností těchto objektů, které jsou pro uživatele podstatné. Běžná BACnet® zařízení obsahují desítky až stovky objektů, které slouží k ovládání hardwarových prvků i pro interní HVAC aplikace zařízení. Běžný uživatel nepotřebuje ani nemůže znát účel všech těchto objektů a v mnoha případech by mohl napáchat svým počínáním škody v systému, kdyby k těmto objektům přístup měl.

Servisní technik nebo správce budovy proto vytvoří místní konfiguraci pro uživatele aplikace a zpřístupní ji uživatelům. Tato konfigurace bude v podobě xml souboru a je nutné ji jedním z mnoha možných způsobů dopravit do úložiště mobilního zařízení, odkud je následně třeba konfiguraci nahrát přímo do aplikace.

Pro různé účely a oblasti je možné vytvářet pouze částečné konfigurační soubory. Aplikace pak musí být schopna generovat ostatní potřebné informace

na základě informací dostupných.

2.2.1 Plná konfigurace

V ideálním případě bude aplikaci předán úplný konfigurační soubor. Formát xml je zvolen z důvodu čitelnosti, snadnému způsobu zpracování a rozšiřitelnosti tohoto formátu.

V konfiguračním souboru jsou objekty sdružovány do logických objektů. Ty jsou pak v uživatelském rozhraní zobrazeny jako jeden celek, kvůli přehlednosti. Logický objekt má v rámci konfiguračního souboru unikátní identifikační číslo a obsahuje číslo šablony, která je využita pro jeho inicializaci a zobrazení.

Součástí logického objektu jsou pak fyzické objekty, tedy BACnet® objekty. Každý objekt nese identifikační číslo BACnet® zařízení, ve kterém je obsažen, typ objektu a identifikační číslo objektu v rámci zařízení.

V rámci fyzického objektu jsou obsaženy jeho vlastnosti. Každá vlastnost nese číslo typu, jak je definováno v BACnet® standardu (vlastnosti jsou určeny výčtem) a název grafického prvku, který slouží k jeho zobrazování. Dalšími možnými atributy jsou hodnota nastavená správcem systému, která je použita místo skutečné hodnoty vlastnosti, dále příznak, zda je tato náhradní hodnota použita vždy, nebo pouze v případě, že není dostupná hodnota vlastnosti ze zařízení. Dalším nepovinným atributem je příznak, zda je daná vlastnost povelovatelná.

Dalším vnořením je pak handler, reference na třídu, která zpracovává obnovování hodnot a jejich propisování do uživatelského rozhraní. Nese informaci o svém typu a může obsahovat libovolné množství dalších atributů, které definuje každý handler zvlášť.

Další součástí konfiguračního souboru je informace o uspořádání logických objektů do oblastí, označené zde jako POV (point of view - úhel pohledu). POV obsahující všechny objekty konfigurace nemusí být přítomný a bude použit v případě, že není přítomný žádný jiný POV. POV obsahuje název a vnořený seznam objektů, které mají být zobrazeny. V případě, že je nutné uspořádat objekty do stromové struktury, budou POV obsahovat také identifikační číslo, stejně jako logické objekty. V tomto případě nesmí být z jednoho POV referencovány zároveň jiné POV a logické objekty. Jde o opatření zajišťující přehlednost uživatelského rozhraní.

2.2.2 Minimální konfigurace

V některých případech nebude možné zpřístupnit úplný konfigurační soubor pro aplikaci. V oblasti užití aplikace nemusí být možnost mít zprovozněný webový server s vystavenými soubory, nemusí být přítomna osoba která by uživatelům předala konfigurační soubor, nebo nemusí být dostatek místa na konfiguračním médiu (kapacita QR kódů a NFC tagů je značně omezená). Pro takové případy je vhodné zavést minimální formát konfigurace oblasti, který

by bylo možno dopravit právě přes fyzická média jako QR kód nebo NFC tag, který může být přilepen na dostupném místě a uživatel aplikace pomocí něj získá konfiguraci.

Plný konfigurační soubor obsahuje velké množství informací, které je standardně možné získat přímo z BACnet® zařízení. Takový soubor sice umožňuje optimalizaci uživatelské zkušenosti, ale některé informace mohou být v případech, kdy není jiná možnost, vypuštěny a nahrazeny výchozími hodnotami.

Pro úspěšnou konfiguraci oblasti je pro každý objekt třeba obsáhnout toto minimum informací:

- ID BACnet® zařízení,
- ID objektu,
- typ objektu.

Pro další úsporu místa je možné pro několik objektů uvést nejprve společné ID zařízení, ve kterém jsou tyto objekty obsaženy. Všechny ostatní informace pak aplikace rekonstruuje komunikací se zařízením a doplní výchozími hodnotami.

Pro každý podporovaný typ objektu bude existovat sada výchozích hodnot, které se mohou částečně měnit na základě informací získaných ze zařízení.

Sada bude obsahovat výčet vlastností, které je aplikace schopna zobrazit. Při načítání minimální konfigurace pak aplikace dotazuje zařízení a porovnává tento výčet s vlastnostmi, které daný objekt opravdu obsahuje (některé vlastnosti nemusí být přítomny, viz sekci 1.1.1). Průnik těchto dvou množin bude následně promítnut do vygenerovaného konfiguračního souboru.

Pro tyto vlastnosti bude následně určen způsob zobrazování do uživatelského rozhraní a další komunikací se zařízením bude zjištěno, zda jsou které vlastnosti povelovatelné.

2.2.3 Reference konfigurace

Pro rychlé vyvolání již načteného konfiguračního souboru lze využít referenční konfiguraci. Ta obsahuje pouze informaci o aplikaci, která má spracovat událost načtení referenční konfigurace, a identifikační číslo konfiguračního souboru, případně i identifikační číslo konkrétního objektu nebo POV.

Díky nízkým nárokům na paměť je možné takovou referenci uložit do levných NFC tagů nebo do malých QR kódů, které mohou být připevněny na dobře přístupných místech.

Tato reference může být zejména užitečná pro servisní techniky. Technik, který zároveň spravuje několik budov, ve kterých je zaveden HVAC systém využívající protokol BACnet®, může mít v mobilním zařízení zároveň nahrané tisíce objektů a potenciálně stovky konfiguračních souborů. Vhodně umístěné médium obsahující referenční konfiguraci, například NFC tag nalepený přímo

na zařízení, umožní technikovi okamžitě zobrazit podstatné objekty bez toho, aby musel zdlouhavě hledat příslušné objekty v dlouhých seznamech POV a objektů.

2.3 Persistence

2.3.1 Uložení konfiguračních souborů

Platforma android umožňuje dva způsoby uložení souborů: Interní a externí uložiště. Interní uložiště je dostupné vždy, soubory které jsou v něm uloženy jsou dostupné pouze pro aplikaci, která je tam uloží, a při odinstalování aplikace jsou všechny soubory z interního uložiště smazány. Externí uložiště je na druhou stranu dostupné pro další aplikace i uživatele, v případě připojení mobilního zařízení přes USB k počítači může být v některých případech nedostupné, a soubory jsou z něj smazány pouze v případě, že jsou uloženy v umístětní získaném voláním metody *getExternalFilesDir()*.

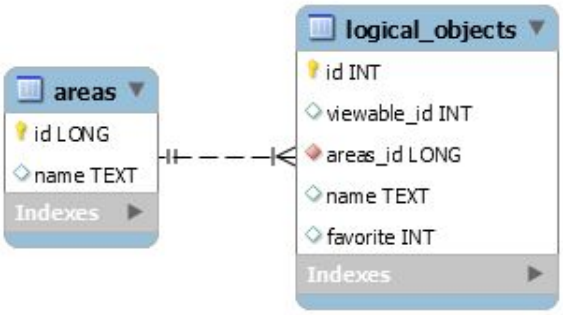
Z důvodů bezpečnosti bylo zvoleno interní uložiště, které je vždy dostupné a uživatel ani jiná aplikace nemůže měnit soubory aplikace BACdroid, ať už úmyslně nebo omylem. Takové změny by mohly způsobit jednak nestabilitu aplikace, ale i možný přístup k objektům, které správce budovy nevystavuje příslušným konfiguračním souborem. Při načtení nového konfiguračního souboru je tento soubor nakopírován do interního uložiště aplikace a zůstává tam, dokud není konfigurace oblasti smazána, nebo dokud není aplikace odinstalována.

2.3.2 Databáze

Uživatel aplikace může přejmenovávat oblasti a objekty, přidávat a odebírat oblíbené objekty. Z důvodu snazší rozšiřitelnosti a povaze oblíbených objektů, které mohou být vybrány zároveň z několika konfiguračních souborů, byla pro tento účel zvolena jednoduchá databáze. Systém Android podporuje snadnou práci s databází typu SQLite. K dalšímu zjednodušení byla použita knihovna ORMLite, která umožňuje mapování relací a dotazů do Java tříd a metod bez zbytečného manuálního vypisování SQLite příkazů.

K používání knihovny ORMLite stačí implementovat dvě jednoduché třídy. Jednou je *OrmLiteConfigUtil*, která zpracovává vytvoření konfiguračního souboru databáze, druhou pak *OrmLiteSqliteOpenHelper*, která zpracovává napojení na databázi samotnou. Umožňuje správu verzí, vytváření dotazů a samotné vytvoření a inicializaci databáze.

Pro jednotlivé tabulky jsou pak vytvořeny třídy a pomocí anotací jsou napojeny do knihovny ORMLite. Udržovány jsou pouze identifikační čísla oblastí a objektů, a měněné hodnoty. Databázové entity jsou znázorněny v obrázku 2.2



Obrázek 2.2: Relační model databáze

2.4 Udržování hodnot, komunikace

Pokud si tak uživatel zvolí, aplikace musí pravidelně občerstvovat hodnoty objektů, které jsou aktuálně zobrazené, i těch, které zobrazené nejsou, ale jsou dostupné na lokální síti. Viditelné objekty je třeba občerstvovat častěji než ty, které nejsou zobrazeny v uživatelském rozhraní. Tyto objekty je ale třeba občerstvovat kvůli zlepšení uživatelského zážitku. Když uživatel prochází oblastmi, nemělo by se mu příliš často stát, že narazí na objekty, u kterých nejsou známy hodnoty jejich vlastností.

2.4.1 Využití knihovny, asynchronní operace

Vzhledem k tomu, že platforma Android nepovoluje provádět síťové operace na hlavním (UI) vlákně, je třeba veškerou práci s knihovnou BACnet4J provádět asynchronně. K těmto účelům je připravena abstraktní třída *AsyncTask* [4], jejímž rozšířením lze snadno jakékoliv časově náročné operace provádět na vedlejším vlákně. Umožňuje provádět operace před vstupem do asynchronního vlákna, při něm, a po něm. Standardně jsou tyto asynchronní úlohy spouštěny sekvenčně. Systém Android sice umožňuje jejich paralelizaci přes explicitní thread pool, pro účely této aplikace je taková funkce ale zbytečná.

Při startu aplikace je inicializován objekt *LocalDevice*, který reprezentuje mobilní zařízení jako BACnet® klientské zařízení v lokální síti. Skrze toto zařízení pak probíhá zpracování všech přímých požadavků na vzdálené BACnet® zařízení. Při jeho inicializaci je vytvářeno napojení na síťovou a transportní vrstvu knihovny, zařízení je následně inicializováno.

Před samotnou komunikací se zařízeními BACnet® je třeba zjistit jejich síťové adresy. Tyto adresy není možné ukládat do konfiguračního souboru, protože se mohou průběžně měnit při dynamickém přidělování adres v lokální síti. Proto je nutné tato zařízení objevit použitím služby `Who-Is`, jak je zmíněno v sekci 1.1.2. Vzhledem k tomu, že velké množství objektů sdílí stejná zařízení, jsou tato zařízení po nalezení ukládána. V případě několika poža-

davků na nalezení stejného zařízení jsou tyto požadavky uloženy a zpracovány zároveň, pouze jednou asynchronní úlohou vyvolávající službu `Who-Is`. Když asynchronní úloha dokončí svou práci, je výsledek předán všem požadavkům a požadavky jsou smazány. Pokud se požadavek hlásí o zařízení, které už bylo nalezeno, dostává odpověď okamžitě. Všechny požadavky jsou ukládány a zpracovány podle identifikačního čísla zařízení, které požadují. Pro ukládání je pro rychlé vyhledání zvolena hašovací tabulka.

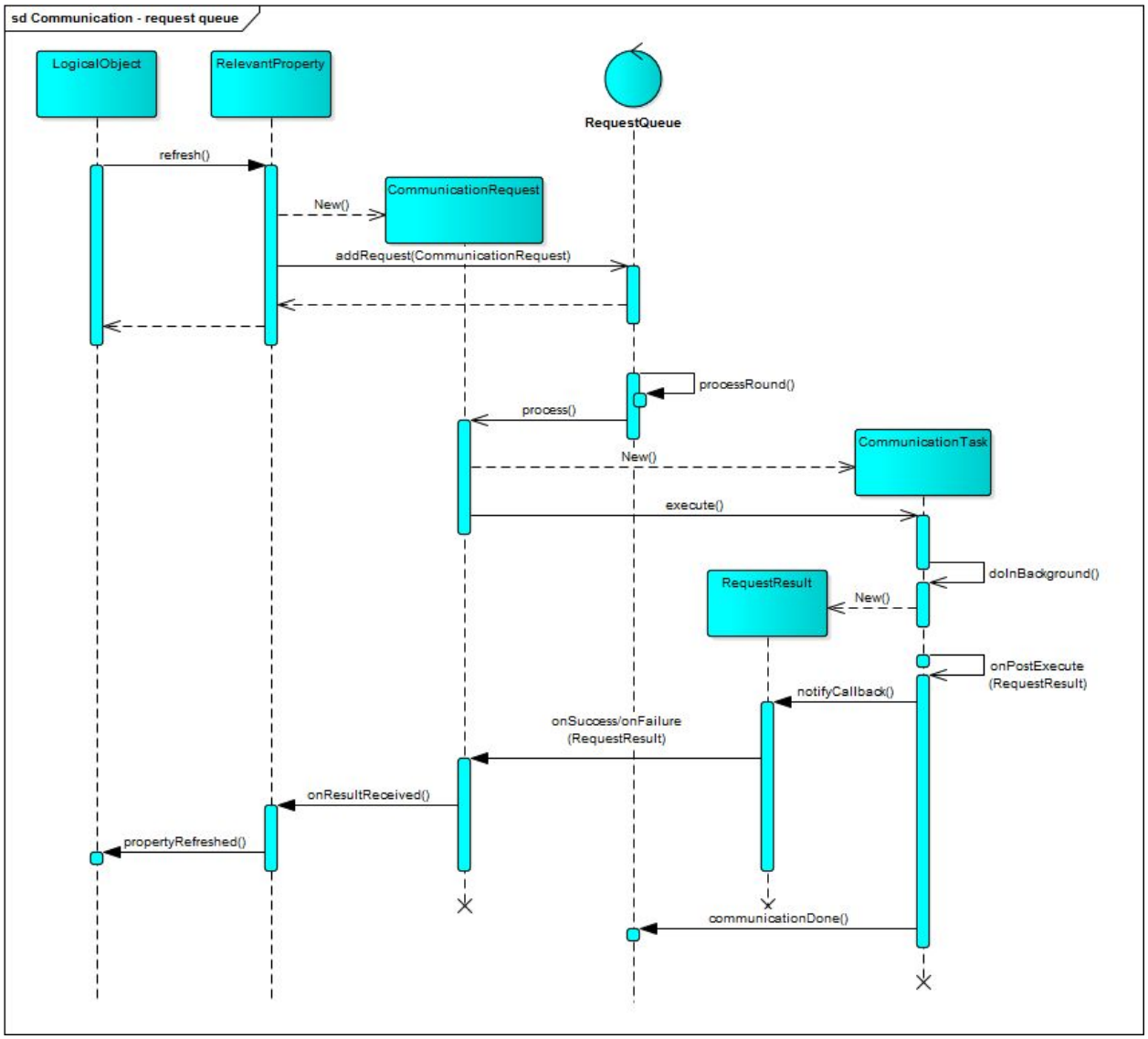
Pro čtení a zápis hodnot vlastností objektů jsou vytvářeny požadavky na komunikaci (objekty třídy *CommunicationRequest*). Tyto požadavky jsou zařazovány do fronty požadavků, která je popsaná v následující sekci. Při zpracování požadavek vytváří asynchronní úlohu (objekt třídy *CommunicationTask* rozšiřující třídu *AsyncTask*). Ta odešle požadavek na vzdálené zařízení, a v závislosti na povaze požadavku může čekat na odpověď ze zařízení. V případě úspěchu i neúspěchu vrací už na hlavním vlákně (tedy po skončení asynchronní práce, metoda *onPostExecute* třídy *AsyncTask*) výsledek jako objekt třídy *RequestResult*, který příslušně ukončí požadavek na komunikaci.

2.4.2 Časování příkazů

Z bezpečnostních důvodů je v aplikaci nutné zavést odmlky v komunikaci do interní sítě BACnet®. Pro větší počet objektů a vlastností jednoho konfiguračního souboru by mohlo dojít k zahlcení sítě a přerušení komunikace mezi BACnet® zařízeními samotnými. Platforma Android umožňuje na výkonnějších zařízeních zaslat několik set požadavků během zlomku sekundy. To by mohlo mít nepříznivé následky na plynulý chod systému ovládání budov. Jako klientská aplikace určená pro běžného uživatele má BACdroid mnohem nižší prioritu než mnohdy mohutný BACnet® systém. Proto je navržena fronta příkazů, která bude požadavky do sítě BACnet® vysílat v dávkách fixní velikosti a mezi zpracovávané dávky vsouvat odmlky.

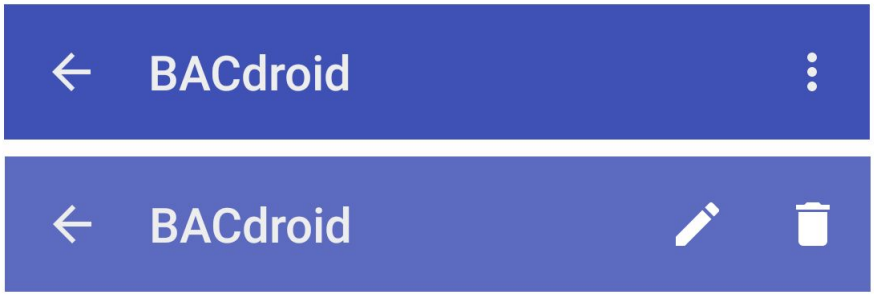
Příkazy jsou do fronty řazeny na základě priority. Priorita příkazů je zavedena z důvodu plynulosti zážitku uživatele z aplikace. Nejvyšší prioritu mají příkazy pro zápis hodnoty, které se musí propsat ideálně ještě před obnovením hodnot vlastností objektů v uživatelském rozhraní. Příkazy čtení hodnot mají prioritu nižší a jsou rozděleny podle toho, zda se týkají viditelných objektů, či objektů na pozadí - viditelné objekty mají při zpracování přednost. Je možné přidávat další hodnoty priority v případě, že bude aplikace rozšířena o další funkce.

Pro časování fronty je využita třída *Handler* [5], která umožňuje jednoduché plánování zpráv podle času. Při vložení nového požadavku do fronty je ověřeno, zda byla fronta prázdná (a tedy Handler neběží). Pokud ano, je zaslána zpožděná zpráva pro handler. V okamžiku přijetí této zprávy Handler začne zpracování jedné rundy požadavků. Jsou nainicializovány a spuštěny asynchronní úlohy pro každý požadavek, a v případě, že ve frontě čekají další požadavky, je opět zaslána zpožděná zpráva pro zpracování. Tím je zajištěno,



Obrázek 2.3: Komunikační diagram zpracování požadavku

že všechny požadavky nebudou odeslány okamžitě po sobě. Když asynchronní úlohy ukončí svoji práci, oznámí tuto událost Handleru i samotnému požadavku, kterému předají výsledek. Tento proces je zjednodušeně znázorněn na obrázku 2.3, kde probíhá obnovení hodnoty vlastnosti logického objektu.



Obrázek 2.4: Lišta nástrojů

2.5 Uživatelské rozhraní

Jako inspirace pro vytvoření uživatelského rozhraní sloužil z velké části Material design společnosti Google [3]. Ten obsahuje doporučení pro aspekty návrhu uživatelského rozhraní jako je navigace mezi prvky aplikace, rozložení prvků, typy grafických objektů i stylování.

2.5.1 Navigační prvky

Nástrojová lišta (Toolbar) je zobrazena v horní části obrazovky a je sdílena ve všech obrazovkách aplikace. V její levé části je zobrazeno navigační tlačítko, nazývané *Up button*, které zprostředkovává akci průchodu na vyšší úroveň obrazovky v navigaci při zanoření. V případě aplikace BACdroid se toto tlačítko chová stejně jako tlačítko zpět, vzhledem k tomu, že neobsahuje takový způsob průchodu, který by přepínal z jedné obrazovky do obrazovky stejné úrovně.

Toolbar dále obsahuje nadpis, který uživateli zjednodušuje orientaci v hierarchii obrazovek. Na domovské obrazovce je zde zobrazen název aplikace, při zanoření potom název oblasti nebo objektu, který je momentálně zobrazen. Nadpis je zobrazen přímo vpravo od tlačítka *Up*.

V pravé části lišty nástrojů jsou zobrazeny dostupné akce. Zobrazuje se zde dropdown menu pro přechod k nastavení a informacím o aplikaci, a v různých místech průchodu aplikací příslušné akce. Při zobrazení přehledu objektů může být viditelná akce obnovení hodnot, při výběru oblasti pak přejmenování a smazání. Příklady lišty nástrojů jsou zobrazeny na obrázku 2.4, změna barvy znázorňuje akci vybraného prvku.

Záložky (Tabs) jsou hlavním navigačním prvkem aplikace, zobrazeny na domovské obrazovce a sloučené s nástrojovou lištou do jednoho grafického prvku. Záložky byly zvoleny z důvodu malého množství obrazovek stejné důležitosti a zobrazení, které je vždy na očích (narozdíl od vyjížděcí navigační lišty). Symbolizují nejvyužívanější aspekty aplikace, z kterých je možno zano-

řit se hlouběji do obrazovek aplikace. Záložky jsou při přechodu automaticky animovány, přechod je umožněn kliknutím na název záložky nebo tažením ze strany na stranu. Aktuálně vybraná záložka je graficky znázorněna barevnou linkou a změnou barvy textu.

Při přechodu na obrazovky nižší úrovně je lišta záložek skryta, zůstává zobrazena pouze nástrojová lišta. Tím se zamezí zmatení z nadpisů záložek odpovídajících jiným obrazovkám, a uvolní se více místa pro zobrazení objektů aplikace. Skrytí lišty je automaticky animováno při odchodu z hlavní obrazovky, a její zobrazení probíhá při přechodu zpět.

Zpětná navigace umožní vrátit se při průchodu aplikací vždy o obrazovku zpět. Lze to provést jak tlačítkem zpět mobilního zařízení, tak tlačítkem *Up* v liště nástrojů. V případě, že je aktuálně zobrazena domovská obrazovka aplikace, je touto akcí aplikace zavřena. Touto akcí je také možné zrušit výběr prvku, který je vyvolán dlouhým podržením (například záznamu oblasti).

Akční plovoucí tlačítko (Floating Action Button) je v aplikaci využito pro přidávání konfigurací oblastí. Je zobrazeno přes seznam již nahraných oblastí a při přechodu na jiné obrazovky dochází k jeho skrývání. Při návratu na obrazovku oblastí je tlačítko opět zobrazeno. Skrytí i zobrazení je animováno. Tlačítko je možné zobrazit i v jiných obrazovkách, pokud k tomu bude pádný důvod. V současné verzi aplikace to není potřeba.

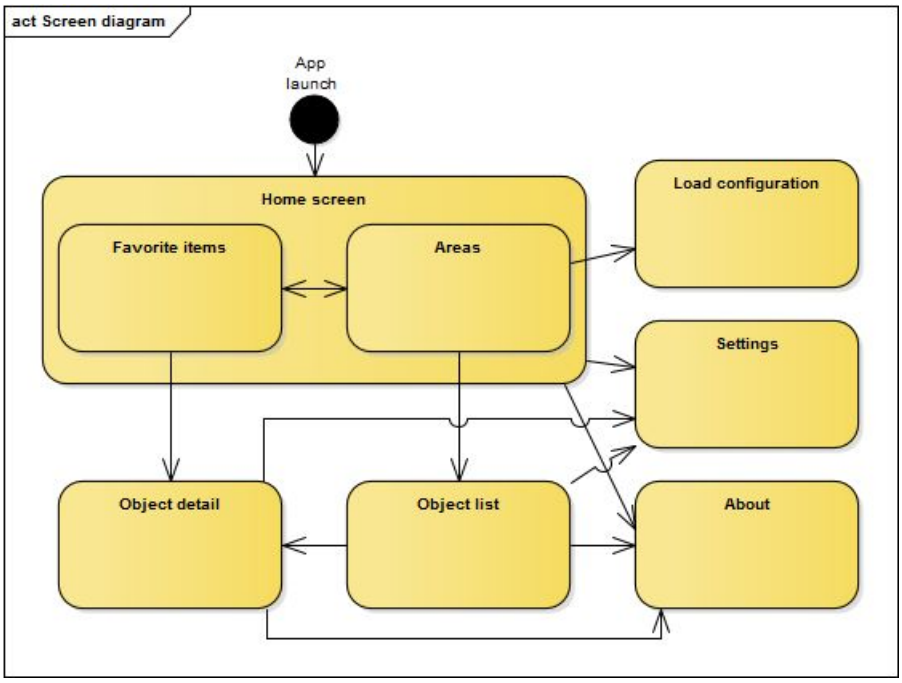
Průchod obrazovkami aplikace je znázorněn na diagramu 2.5. Znázorněna je domovská obrazovka, která obsahuje dvě záložky. Na seznam objektů lze přejít výběrem oblasti, a z něj, a také z obrazovky oblíbených položek, lze přejít na detail objektu. Z obrazovky oblastí je plovoucím tlačítkem umožněn přechod na načtení nové konfigurace. Ze všech obrazovek lze díky menu v liště nástrojů možné zobrazit nastavení a informace o aplikaci.

2.5.2 Zobrazení objektů

Vzhledem k různorodosti BACnet® objektů je třeba vytvořit prvky uživatelského rozhraní, které jsou na jednu stranu dobře rozšiřitelné a mohou pojmout mnoho odlišných částí, ale které zároveň vytvářejí pocit celistvosti a jednoty při zobrazení v seznamech.

2.5.2.1 Karta a detail

Pro zobrazení hlavních informací o objektech v seznamu byl zvolen grafický prvek karta (*Card*). Ten je vhodný pro složité, různorodé objekty které umožňují provádět několik akcí. Pro větší obrazovky je možné je uspořádávat do několika sloupců i v případě, že se podstatně liší ve velikosti.



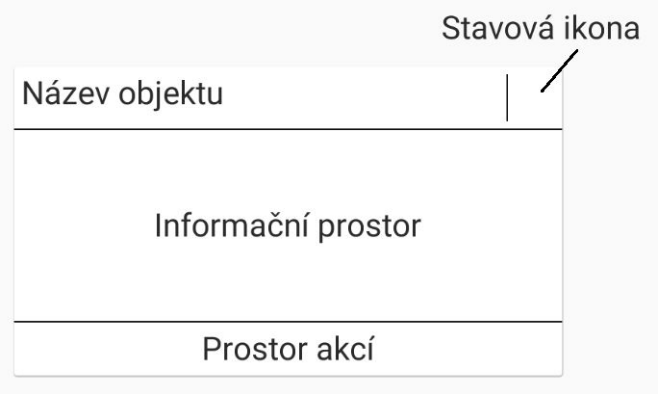
Obrázek 2.5: Diagram průchodu obrazovkami aplikace

Karta každého objektu obsahuje hlavičku, která se skládá z názvu objektu a stavové ikony, která může zobrazovat probíhající operace na pozadí a varování a chyby způsobené problémy při komunikaci s BACnet® zařízeními.

Pod hlavičkou je prostor pro zobrazení informací o objektu, který je vyplňován podle potřeby a dostupných vlastností. Je možné zde zobrazit téměř jakýkoliv prvek, krom některých, které by mohly způsobit kolize při ovládání, například se záložkami hlavní obrazovky (horizontální osa je určena pro změnu záložek, prvky jako posuvník proto není vhodné uvnitř karet použít).

Ve spodní části karty jsou pak zobrazeny akce, které je možné s objektem provádět. Tlačítko pro přechod na detail a tlačítko pro přidání či odebrání oblíbené položky je dostupné vždy, navíc lze ale zobrazit další tlačítka podle potřeby jednotlivých objektů, například může být zobrazeno tlačítko pro nastavení výchozí hodnoty objektu (operace `Relinquish`, viz 1.1.3). Tyto součásti karty jsou znázorněny na obrázku 2.6.

Pro detail objektu aplikace přechází do zvláštní obrazovky, která se rozložením příliš neliší od karty objektu. Zde je ale prostor pro zobrazení více informací, případně dalších BACnet® objektů svázaných s daným logickým objektem. Tyto informace by bylo možné zobrazit i v kartě, ale při procházení seznamu většího množství objektů by tyto informace mohly být matoucí, došlo by ke snížení počtu zároveň viditelných objektů, seznam by narostl na vertikální velikosti.



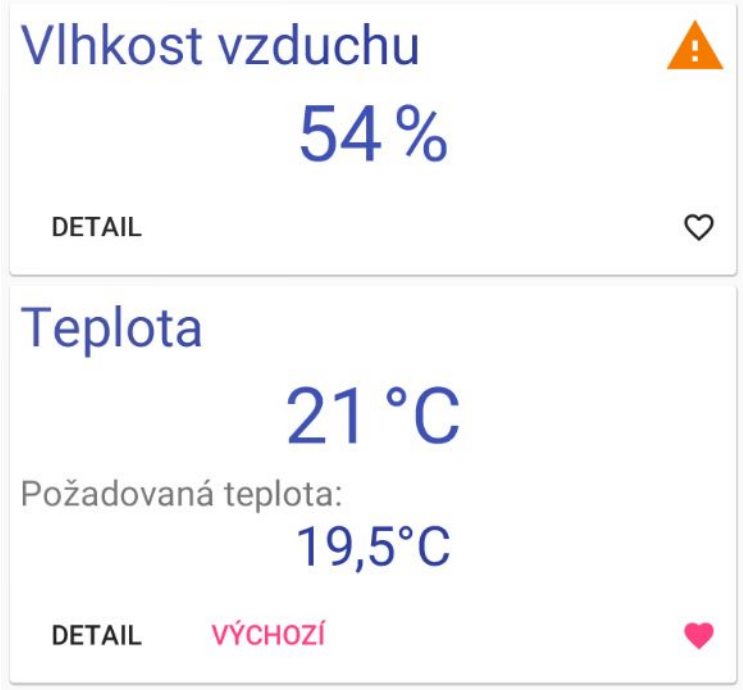
Obrázek 2.6: Rozložení komponent karty objektu

2.5.2.2 Typy logických objektů

Prostý logický objekt je složen pouze z jednoho fyzického BACnet® objektu. Při zobrazení karty tedy informační prostor slouží pouze pro prvky tohoto objektu. Při zobrazení detailu je informační prostor rozšířen o hodnoty dalších vlastností příslušného BACnet® objektu.

Složený logický objekt se skládá z několika fyzických BACnet® objektů. Při zobrazení karty mohou, ale nemusí být součástí informačního prostoru všechny tyto objekty. Některé mohou být zobrazeny například pouze v detailu objektu, aby grafický prvek příliš nenarůstal v seznamu. I tak ale karta může zobrazit složení několika fyzických objektů, a tedy několik hodnot, do jednoho celistvého grafického prvku.

Příkladem takového složeného objektu může být teplota v místnosti. Hlavní část informačního prostoru zabírá současná hodnota teploty v místnosti, která může být reprezentována například objektem `Analog Input`. Pod touto informací bude v menším fontu vyvedena hodnota požadované teploty, včetně popisku, co tato hodnota znamená. Umožní přímo z objektu karty nastavit požadovanou teplotu a pozorovat rozdíl oproti skutečné teplotě. Při zobrazení detailu pak bude odkryt další objekt, například `Trend Log`, v podobě grafu hodnot teploty v místnosti za určitou dobu. Tento graf by byl příliš veliký pro zobrazení zároveň s hodnotami současné teploty a požadované teploty. Příklady jednoduchého a složeného objektu zobrazeného v kartě jsou na obrázku 2.7



Obrázek 2.7: Ukázky karet objektů

Kapitola 3

Realizace

3.1 Aktivita aplikace

Implementace aplikace BACdroid využívá jedné aktivity, která slouží k udržování aktuálně používaných dat o objektech a oblastech. Jedna aktivita byla zvolena zejména z důvodu udržování objektu třídy *LocalDevice*, který je napojený na síťové rozhraní a není možné ho snadno předávat mezi aktivitami. Aktivita dále udržuje napojení na databázi aplikace, zprostředkované třídou *SQLiteHelper*.

Při startu aktivity dochází k načtení všech konfigurací uložených v interním uložišti aplikace a inicializaci uživatelského prostředí. Jednotlivé obrazovky jsou reprezentovány třídami rozšiřujícími třídu *Fragment*. Aplikace je v současné verzi navržena pouze na mobilní telefony, rozhraní není optimalizováno pro tablety. Díky fragmentům je taková optimalizace ale snadná, stačí pouze do hlavní aktivity aplikace přidat víc fragmentů zároveň, v závislosti na velikosti obrazovky zařízení, a orientaci.

Hlavní aktivita udržuje stav aplikace, jako například informaci o tom, jestli je vybrán nějaký prvek (v tomto stavu je třeba upravit chování tlačítek zpět a *Up* a podbarvit akční lištu). Dalším sdíleným prvkem je plovoucí akční tlačítko, které je sice zatím využito pouze v jednom fragmentu, ale v dalších verzích se může vyskytnout i v jiných. V aktivitě je referencováno z důvodu plynulosti animací při přechodech mezi fragmenty.

Aktivita aplikace je využita i pro animace lišty záložek, která je skrývána při opouštění hlavní obrazovky a odkrývána při návratu. Informace, zda se má lišta záložek skrýt, je zpracována fragmenty, které následně volají příslušnou metodu aktivity.

3.1.1 Fragmenty

V platformě Android je fragment brán jako součást aktivity, reprezentující část uživatelského rozhraní. Lze skládat více fragmentů do jedné aktivity, pří-

padně sdílet fragment mezi několika aktivitami. Má svůj vlastní životní cyklus, který je jen částečně závislý na cyklu aktivity [2]. V mobilních telefonech je běžné, že je v jeden okamžik zobrazen jeden fragment, zatímco na tabletech jich obrazovku sdílí zároveň několik. V současné verzi aplikace, která je optimalizované pouze pro malé obrazovky mobilních telefonů, jeden fragment reprezentuje jednu z obrazovek, které jsou zmíněné na diagramu 2.5.

Jednotlivé fragmenty se na aktivitu napojují v rámci svého životního cyklu pomocí několika rozhraní. Tato rozhraní umožňují fragmentům volat metody aktivity. V případě, že je nutné předávat určité události z aktivity fragmentům, například vytváření ikon akcí v liště nástrojů je v určitých případech závislé na fragmentu, je využit *FragmentManager*, který umožní nalézt aktuálně používaný fragment.

Přechody mezi fragmenty jsou řešeny pomocí třídy *FragmentManager*, která tyto události zpracovává jako transakce. Všechny transakce jsou přidány na takzvaný *back stack*, který umožňuje jednoduchou správu navigace zpět v průchodu aplikace.

Všechny fragmenty aplikace rozšiřují třídu *BaseFragment*, která je vytvořena pro sjednocení napojení fragmentu na hlavní aktivitu. Obsahuje také výchozí metody pro určité stavy fragmentů (informaci zda se mají skrýt záložky, standardní vzhled akčních ikon v nástrojové liště). Tyto metody je v případě potřeby možné přetížit.

Pro účel záložek jsou nutné vnořené fragmenty, které jsou v platformě Android dostupné od verze 4.2, ale práce s nimi je v nižších verzích umožněna díky podpůrným knihovnám pro zpětnou kompatibilitu. To značně zjednodušuje správu uživatelského rozhraní v aktivitě aplikace, která by jinak musela prohazovat grafické prvky v rozložení, což může vést k zátěži a následnému zpomalení chodu aplikace.

3.2 Implementace objektů

Pro zpracování konfiguračních souborů xml byl zvolen *ExpatPullParser*, implementující rozhraní *XmlPullParser*, který je doporučený pro použití na platformě Android. Při procházení lze porovnávat aktuální tag, číst z něj atributy a kontrolovat, zda jde o otevírací či ukončovací tag. Celý proces parsování je implementován třídou *XmlDefinitionParser*, která z přečtených souborů vytváří instance objektů třídy *AreaConfiguration*.

Po načtení konfigurace ze souboru je třeba sestavit oblast a její objekty. K tomu je hojně využíván návrhový vzor builder a továrna. Návrhový vzor továrna je zejména praktický pro snadné rozšiřování aplikace o další typy objektů bez toho, aby bylo nutné příliš zasahovat do procesu sestavování. V závislosti na typu BACnet® objektu továrna vrací příslušný *RemoteObjectWrapper*. Pro napojení na uživatelské rozhraní je také využita továrna, která vytváří objekty třídy *ViewHandler*, které jsou popsány níže. Návrhový vzor builder na dru-

hou stranu umožňuje nastavit velké množství defaultních hodnot, které není v některých případech možné do konfiguračního souboru vyplnit (viz kapitolu 2.2).

3.2.1 Zpracování hodnot

Protokol BACnet® popisuje velké množství datových typů, které mohou vlastnosti objektů mít. Ne vždy je ovšem snadné takové hodnoty zobrazit do uživatelského rozhraní systému Android. Z toho důvodu byl vytvořen systém tříd *ViewHandler*, které zpracovávají převod těchto datových typů do typů vhodných pro zobrazení, a zprostředkovávají jejich propsání do jednotlivých grafických elementů.

Vzhledem k různorodosti datových typů i možností zobrazení může potenciálně vzniknout velké množství typů a podtypů těchto handlerů. Při načítání z xml konfigurace není možné jednotně definovat případné potřebné vstupní atributy pro konstrukci těchto objektů. Proto je vytvořena metoda umožňující handleru, který byl již vytvořen, přečíst si svoje atributy přímo z xml parseru v průběhu načítání.

Při implementaci dalších handlerů je vhodné uvnitř každého z nich vytvořit metody, které zpracovávají převod mezi datovými typy protokolu BACnet®, override hodnotami a hodnotami, které je možné zobrazit v uživatelském rozhraní.

Příkladem může být *BinaryHandler*, který převádí výčtový datový typ `BinaryPV` na *boolean*, pomocí kterého mění stav tlačítka *CompoundButton* (dvoustavové tlačítko, v systému Android je několik typů). Obsahuje také metodu na převod stavu tlačítka zpět do typu `BinaryPV` pro případ, že je třeba hodnotu odeslat do BACnet® zařízení jako povel.

3.2.2 Implementované BACnet® objekty

V aplikaci BACdroid je implementováno několik základních objektů:

- `Analog Input`,
- `Analog Value`,
- `Binary Output`,
- `Binary Value`,
- `MultiState Value`.

Tyto objekty zobrazují několik základních vlastností pro které jsou připravené handlery. Objekty analogové hodnoty a vstupu jsou zobrazeny stejně, hlavní rozdíl je v možnosti zadávání hodnoty a povelování BACnet® zařízení, které u vstupního objektu podporováno není. Binární objekty jsou prakticky

totožné, povelovatelnost závisí na konfiguračním souboru. Pro zkoušení na živém zařízení je ale `Binary Output` zajímavější, lze pomocí něj přímo povelovat výstupní hardwarové zařízení, například žárovku.

K implementaci jsou připravené objekty `Schedule` a `Trend Log`, u kterých došlo k otestování čtení hodnot ze zařízení, ale z časových důvodů (práce s těmito objekty je poměrně náročná) nedošlo k jejich plné implementaci do uživatelského rozhraní.

3.2.2.1 Zadávání hodnot

Pro povelování objektů BACnet® zařízení bylo vytvořeno několik grafických prvků. Nejjednodušším je povelování binárních objektů. Pro ty stačí do karty nebo detailu objektu vložit stavové tlačítko (například přepínač) a odchytávat změny jeho stavu. Sloužit může zároveň k zobrazení současné hodnoty i k povelování zařízení.

Pro analogové a vícestavové objekty je pro nastavení hodnoty použito dialogové okno. To umožňuje zadat hodnotu a potvrdit nebo zrušit zadání. V případě vícestavového objektu existují dvě varianty dialogového okna, v závislosti na počtu možných stavů objektu. Pokud má objekt méně než pět stavů, je výběr umožněn skupinou radio tlačítek. V případě pěti a více stavů je použito rolovací menu. Tento výběr souvisí s přehledností a úsporou místa pro menší displeje.

3.2.2.2 Stav objektu

Stav objektu je znázorněn stavovou ikonou v pravém horním rohu karty nebo detailu objektu. Objekt může mít několik stavů, které se mohou překrývat, vždy je však zobrazen pouze jeden:

- normální stav,
- stav obnovování hodnot,
- stav varování,
- chybový stav.

Při normálním stavu je ikona objektu prázdná. Pokud je aktivní jiný stav, zobrazí se příslušná ikona. Pokud je aktivní pouze stav varování, je zobrazena ikona varování. Pokud je aktivní chybový stav a aplikace neaktualizuje hodnoty objektu, je zobrazena ikona chyby. Pokud dochází k obnově hodnot vlastností objektu, je vždy zobrazen progress bar.

Kliknutím na ikonu chyby nebo varování je možné vyvolat dialogové okno obsahující informace o současném stavu objektu. Pokud je zároveň aktivní stav varování i chyby, je zobrazena pouze ikona chyby, ale v dialogovém okně jsou v seznamu vypsány všechny důvody chyb i varování, včetně příslušných ikon, pro rozlišení závažností stavů.

3.3 Přidání dalších objektů

Pro přidání implementace dalších BACnet® objektů je třeba vykonat několik zásadních kroků v různých částech kódu aplikace.

Pro každý nový implementovaný BACnet® objekt musí vzniknout třída rozšiřující třídu *RemoteObjectWrapper*, která zpracovává jeho zobrazení do logického objektu. Továrna *RemoteObjectFactory* musí být rozšířena tak, aby při zadání typu tohoto objektu vracela příslušný wrapper.

Pro každou vlastnost objektu, která je nově přidávána, je třeba vytvořit odpovídající *ViewHandler* a upravit továrnu *ViewHandlerFactory* tak, aby vracela příslušný handler podle jeho identifikačního čísla, které se ukládá do konfiguračního souboru. Handler si musí být schopen načíst potřebné parametry z xml parseru, pokud jsou tyto parametry třeba.

Každý nový objekt musí implementovat své zobrazení do uživatelského rozhraní. Rozložení grafických prvků v detailu objektu, a v kartě, pokud se objekt do karty zobrazuje. Pro objekt `Schedule` by bylo vhodné použít grafický prvek podobný zobrazení týdne v aplikaci Google Calendar. Pro objekt `Trend Log` pak využít grafickou knihovnu, která umožňuje zobrazovat grafy.

Zadávání hodnot do složitějších objektů je záležitost budoucích implementací. Při zadávání je třeba ověřovat, že jsou hodnoty zadány tak, aby je bylo možné bez chyb propsat do BACnet® zařízení.

3.3.1 Skládání objektů

Pro vytváření složených logických objektů je třeba rozhodnout, jak bude prováděn výběr hlavního objektu (toho, jehož jméno bude využito pro pojmenování logického objektu). Do konfiguračního souboru je také třeba začlenit příznaky pro fyzické objekty, které určí, zda budou příslušné objekty zobrazeny v kartě, nebo pouze v detailu. Pořadí při zobrazení objektů v kartě bude dáno jejich pořadím v konfiguračním souboru.

Složené logické objekty zatím nejsou implementovány. Rozšířením třídy *LogicalObject* toho lze docílit poměrně snadno, rozšiřující třída může ze seznamu fyzických objektů vybrat ty, které na sebe navazují, propojit jejich hodnoty (například zobrazení jednotek `Trend Log` objektu se provádí přes logovaný objekt, který je uložený v objektu `Trend Log` jako reference). Pro zobecnění složeného logického objektu lze využít například návrhového vzoru mixin - ten by byl aplikován například při obsažení objektu `Schedule`, který by byl zobrazen pouze v detailu objektu. Mixin by měl na starost získat si od logického objektu příslušné informace a reagoval by na události, jako například přechod na detail, vykreslováním potřebných prvků do grafického rozhraní.

Kapitola 4

Testování

4.1 Testy na živých zařízeních

Aplikace BACdroid byla v průběhu implementace testována oproti skutečným BACnet® zařízením, které byly nakonfigurovány pro účely vývoje. Z technických důvodů nebylo možné aplikaci testovat v systému, který by zároveň ovládal inteligentní budovu. Zařízení, které bylo zapůjčeno společností Johnson Controls je ovšem stejné, jako zařízení skutečně využívaná v produkci.

Zapůjčené zařízení NCE2566 je složeno ze dvou BACnet® zařízení, z nichž jedno se chová jako síťový prvek vnější sítě. Druhé zařízení je k němu vnitřně připojeno a je zanořeno v interní BACnet® síti. Zařízení má velké množství hardwarových vstupů a výstupů, konektor pro Ethernet, RS232, a dále displej s tlačítky, umožňující přímý monitoring a správu objektů.

Pro konfiguraci objektů a ověřování práce s nimi byl využit dedikovaný desktopový nástroj, který umožňuje pokročilou manipulaci se zařízením i jeho objekty.

Další užitečný nástroj je součástí aplikace BACdroid. Tento nástroj se nebude vyskytovat v produkční verzi (běžný uživatel nesmí mít přístup k informacím o objektech, které nemá přiděleny v konfiguračním souboru). V aplikaci BACdroid je tento nástroj zobrazen jako třetí záložka hlavní obrazovky a umožňuje provádět objevování všech dostupných zařízení a jejich objektů pomocí služeb `Who-Is` a `ReadProperty` (zařízení mají vlastnost `Object_List`). Tento nástroj byl vytvořen pro ověření práce knihovny i pro porovnání funkčnosti ostatních prvků aplikace a může být dále využit v super-user verzi aplikace.

Při samotném testování knihovny BACnet4J a jejím využití pro chod aplikace byly nejvíce zkoumány různé aspekty práce přes síť. Přestože aplikace rozhodně není v současné verzi stoprocentně stabilní, mnoho problémů bylo odhaleno a zvládnuto. Hlavním problémem bylo řešení stavu, kdy mobilní zařízení není připojeno k WiFi síti, která je součástí konfigurace oblasti. V tomto případě je uživatel na tuto skutečnost upozorněn, a pokud je to možné, apli-

kace se sama pokouší připojit k této síti. Tím dochází k zamezení vznikání chyb souvisejících s nedostupnými zařízeními a objekty.

Dalším krokem, v okamžiku kdy už je mobilní zařízení připojeno ke správné síti, je pak ověření nalezení zařízení a příslušných objektů. V případě jakýchkoliv chyb se síťovou komunikací je uživatel na tuto skutečnost upozorněn stavovou ikonou, která byla přidána právě z tohoto důvodu - při snaze o obnovení nedostupných zařízení je uživateli doporučeno zkontrolovat, zda je připojen ke správné síti a akci opakovat.

Čtení hodnot implementovaných objektů bylo testováno jak pomocí discovery nástroje, tak samotnými implementovanými třídami objektů. Výjimečně se stává, že vyprší časový limit žádosti - tomu se zabránit nedá a vyšší časový limit jen způsobuje zbytečně dlouhé čekání. Uživatel je na tento stav upozorněn a při opakování pokusu dochází ke správnému čtení hodnot.

Při zapisování hodnot do objektů a jejich povelování byla nalezena chyba v knihovně BACnet4J. Narozdíl od čtení hodnot je potvrzovací zpráva zápisu hodnoty jiného typu, jde o prostý acknowledgement. V případě jeho přijetí ale dochází k jeho nesprávnému zpracování synchronní metodou čekání na výsledek požadavku. Přestože je odpověď obdržena s dostatečnou časovou rezervou, metoda vyhazuje výjimku, jako by se jednalo o vypršení časového limitu žádosti. Tato skutečnost byla nahlášena tvůrci knihovny, zatím ale nebyla vydána nová verze, která by tuto chybu opravovala.

Součástí testování byly i zatím nepodporované objekty `Schedule` a `Trend Log`. Došlo k úspěšným pokusům o čtení hodnot jejich podstatných vlastností. Čtení objektu Schedule je jednodušší, protože lze použít službu ReadProperty. Objekt při dotazování na hodnotu vlastnosti běžného časového plánu vrací složitý objekt, který reprezentuje týdenní časový plán. Pro čtení trendovaných hodnot je nutné využít službu `ReadRange`, která byla také otestována, ovšem pouze na omezeném vzorku hodnot (čtení proběhlo pro deset záznamů). Pro implementaci objektu `Trend Log` je nutné vyřešit velké množství možností způsobů čtení a ověřit také způsob, jakým knihovna BACnet4J řeší segmentaci zpráv, ke které může dojít při čtení velkého množství hodnot zároveň.

Testování ukázalo, že implementované objekty pracují ve většině případů podle očekávání a že až na drobné výjimky je knihovna BACnet4J spolehlivá. Je nutné dořešit a rozdělit velké množství různých typů problémů a odděleně o nich předávat informace uživateli.

4.2 Uživatelské testy

Vzhledem k orientaci aplikace na běžné uživatele bylo třeba provést testy uživatelského rozhraní. Testování bylo podrobeno několik osob, z nichž některé byly znalé protokolu BACnet® a systémů HVAC, některé bez těchto znalostí. Takto zvoleným vzorkem je zajištěno pokrytí zkušeností jak ze strany osob, které budou případně využívat pozdější verzi aplikace, tak ze strany těch, kteří

budou pracovat na vytváření konfigurací jako součást zakázky automatizace budov.

4.2.1 Testovací konfigurace

Pro účely uživatelských testů je vytvořena konfigurace BACnet® zařízení, sada konfiguračních souborů a několik objektů, které slouží k účelům těchto testů. V testovacích scénářích budou využity následující prvky:

- WiFi router napojený na BACnet® zařízení NCE,
- mobilní telefon Huawei P8,
- tři konfigurační soubory, z nichž jeden je v aplikaci již načten, dva jsou připraveny ve složce Downloads v uložišti mobilního zařízení,
- NFC štítek s připraveným identifikačním číslem jedné z konfigurací,
- několik BACnet® objektů různých typů obsažených v konfiguračních souborech.

4.2.2 Testovací scénáře

Pro uživateslké testy bylo připraveno několik testovacích scénářů. Tyto scénáře pokrývají akce, které nemusí být naprosto zřejmé. Ohled musí být brán také na to, že aplikace využívá principů Material designu, který není všeobecným standardem pro aplikace na platformě Android, a v jistých scénářích uživatelé nemusí znát určité ovládací prvky.

Načtení konfiguračního souboru

1. Přejděte na záložku oblastí,
2. stiskněte tlačítko pro přidání oblasti (akční tlačítko),
3. vyberte konfigurační soubor kancelar.xml ze složky Download v uložišti zařízení,
4. ověřte načtení nové konfigurace.

Zobrazení oblasti načtením NFC tagu

1. Přiložte mobilní telefon k NFC štítku,
2. počkejte dokud nedojde k načtení,
3. ověřte zobrazení oblasti Pokoj.

Obnovení hodnot zobrazených objektů

1. Manuálně obnovte hodnoty všech zobrazených objektů (ikona v liště nebo gesto stažení),
2. vyčkejte, dokud se hodnoty neobnoví,
3. ověřte načtené hodnoty.

Manuální zobrazení objektů v oblasti

1. Přejděte zpět na záložku oblastí,
2. klikněte na oblast Kancelář, kterou chcete zobrazit,
3. ověřte zobrazení objektů oblasti Kancelář.

Nastavení analogové hodnoty

1. V seznamu objektů najděte objekt Požadovaná teplota,
2. klikněte na hodnotu požadované teploty,
3. zadejte hodnotu požadované teploty 19,5°C,
4. potvrďte zadání hodnoty,
5. vyčkejte na obnovení hodnoty objektu.

Zobrazení detailu objektu

1. V seznamu objektů najděte objekt Teplota,
2. zobrazte detail objektu (tlačítko detail),
3. ověřte zobrazení detailu objektu.

Přejmenování oblasti

1. Přejděte zpět na seznam oblastí,
2. dlouze podržte položku oblasti Pokoj,
3. klikněte na ikonu přejmenování (tužka),
4. zadejte nový název místnosti - Ložnice a potvrďte přejmenování,
5. ověřte přejmenování oblasti.

Přidání položky do oblíbených položek

1. Přejděte do oblasti Kancelář,
2. kliknutím na ikonu srdce přidejte objekt Světla do oblíbených objektů,
3. vraťe se zpět na hlavní obrazovku a přejděte do záložky oblíbených objektů,
4. najděte v seznamu objekt Světla, který jste přidali.

Odebrání položky z oblíbených položek

1. V záložce Oblíbené odebrete objekt kliknutím na ikonu srdce,
2. ověřte odebrání z oblíbených položek.

4.2.3 Poznatky z testů

Testům bylo podrobeno několik osob, zjištěn byl věk, zkušenosti s platformou Android a znalost protokolu BACnet®.

Tester 1

- **Pohlaví:** můž
- **Větk:** 51 let
- **Preferovaná mobilní platforma:** Android
- **Zkušenosti s protokolem BACnet®:** pokročilé znalosti

Tester si všiml, že při zadávání hodnoty pro `Analog Value` umožňuje příslušné textové okno zadat desetinnou tečku, nikoliv čárku, přestože jazyk telefonu je nastaven na Češtinu. Dále mu vadilo, že by se v seznamu oblíbených objektů mohly vyskytnout dva stejně pojmenované objekty z různých oblastí. Bylo by tedy vhodné jako součást prvku zobrazit i název oblasti. Uživatel také upozornil na zobrazení tlačítka *Up* na hlavní obrazovce, kde se chová jako ukončovací tlačítko a může mást, uživatel si myslel že lze jít v průchodu aplikace ještě zpět.

Tester 2

- **Pohlaví:** žena
- **Větk:** 40 let
- **Preferovaná mobilní platforma:** Android

- **Zkušenosti s protokolem BACnet®:** bez znalostí

Testerka při přejmenování oblasti omylem zadala na konec názvu několik prázdných řádků, což způsobuje nevzhledný efekt při zobrazení v seznamu. Ikona pro přidání a odebrání oblíbeného objektu se zdá být přiliš malá, uživatelce se podařilo přidat oblíbenou položku až po několika kliknutích.

Tester 3

- **Pohlaví:** můž
- **Větk:** 32 let
- **Preferovaná mobilní platforma:** iPhone
- **Zkušenosti s protokolem BACnet®:** pokročilé znalosti

Uživateli chyběla při odebírání položek ze seznamu oblíbených objektů možnost vrátit akci zpět, podobně jako je umožněno u odebírání oblastí.

Tester 4

- **Pohlaví:** žena
- **Větk:** 51 let
- **Preferovaná mobilní platforma:** iPhone
- **Zkušenosti s protokolem BACnet®:** bez znalostí

Uživatelka neměla k ovládání aplikace žádné výhrady, ale vyskytl se stejný problém s ikonou oblíbené položky.

Tester 5

- **Pohlaví:** můž
- **Větk:** 23 let
- **Preferovaná mobilní platforma:** Android
- **Zkušenosti s protokolem BACnet®:** bez znalostí

Uživatel si všiml, že v seznamu objektů v oblasti chybí možnost přejmenování oblasti a v detailu objektu možnost obnovení hodnot. Také by očekával, že se hodnoty objektů začnou obnovovat hned po zobrazení.

4.2.4 Vyhodnocení testů

Testy proběhly veskrze úspěšně, dokonce i uživatelé, kteří nejsou zvyklí na platformu Android se v aplikaci rychle orientovali. Nebyla pozorovaná žádná závislost znalostí protokolu BACnet® na schopnostech uživatele používat aplikaci, což je velmi pozitivním zjištěním. Vzhledem k uživatelským testům a konstruktivním připomínkám testerů bylo v aplikaci provedeno několik změn.

- Tlačítko *Up* je na hlavní obrazovce skryto. Pro zavírání se v Android aplikacích běžně používá tlačítko zpět, není tedy nutné přidávat alternativu.
- Pro zadávání názvu oblasti při přejmenování bylo vstupní pole omezeno na jeden řádek. V případě zadávání hodnoty s desetinnými místy pro analogový objekt se jedná o chybu ze strany systému Android, který nepočítá s desetinnou čárkou pro určité lokalizace, na místo desetinné tečky.
- Při odebrání oblíbené položky byla přidána možnost vrátit akci zpět, implementovaná pomocí třídy *SnackBar*.
- Pro ikonu pro přidání nebo odebrání oblíbeného objektu byla rozšířena oblast zachytávající událost kliknutí.
- V obrazovce seznamu objektů v místnosti a detailu objektu byly v liště nástrojů přidány nové akce.

Závěr

Zhodnocení vypracování

Splnění zadání

V rámci této diplomové práce došlo k vývoji mobilní aplikace BACdroid a návrhu přidruženého systému pro správu zařízení implementující protokol BACnet®.

1. Byla provedena analýza několika knihoven implementujících protokol BACnet® a vyhodnocením byla vybrána knihovna BACnet4J, která byla následně použita ve vzniklé aplikaci.

2. Vzniklá mobilní aplikace je klientským prvkem připojujícím se do lokální sítě za účelem komunikace s BACnet® zařízeními. Implementuje služby pro vyhledání zařízení a služby pro manipulaci s objekty, určené ke čtení a zápisu hodnot. Součástí aplikace je i implementace několika základních BACnet® objektů představujících analogovu, binární a vícestavovou hodnotu. Možnost implementace komplexnějších služeb a objektů byla nastíněna v předchozích kapitolách a bude dále zmíněna v následující sekci.

3. Uživatelské rozhraní bylo navrženo a implementováno s příhledem k cílové uživatelské skupině a v potaz byla brána další možná rozšíření pro složitější objekty. Verze pro techniky správy budov je diskutována v následující sekci, v této verzi aplikace od ní bylo odhlédnuto vzhledem k diametrálně odlišným požadavkům na škálu funkcí aplikace. Při návrhu uživatelského rozhraní byl hlavní inspirací Material design.

4. Pro způsob konfigurace byly zvoleny konfigurační soubory typu xml, které obsahují informace o dostupných zařízeních i způsobu napojení jejich objektů do uživatelského rozhraní. Pro uživatelské vstupy byla dále využita databáze, která uchovává změny provedené s objekty. Pro

zjednodušení vyvolání konfigurace z paměti telefonu je v aplikaci implementována možnost přečíst NFC štítek obsahující identifikační číslo konfigurace a spustit aplikaci přímo v náhledu na tuto oblast.

5. Aplikace byla testována proti živým BACnet® zařízením i emulátoru a byly provedeny uživatelské testy za pomoci různorodé skupiny potencionálních uživatelů.

Stav aplikace

Mobilní aplikace BACdroid je v současné verzi spíše „proof of concept“, rozhodně není připravena k nasazení. Součástí této práce je ale rozsáhlý popis celého systému. Před nasazením musí dojít k jeho kompletní implementaci. Aplikace zatím slouží jako ukázka možností, jaké skýtá platforma Android pro oblast správy inteligentních budov. Jako takovou lze aplikaci představit potencionálním zájemcům o její další rozvoj.

Pro dokončení aplikace je nutné provést zejména změny ke zvýšení její stability co se týče komplexního životního cyklu aplikací a jejich aktivit v operačním systému Android. Je třeba zajistit obnovu stavu po zabití aplikace systémem (ke kterému může dojít při nedostatku systémových prostředků), rozšířit uživatelské rozhraní o animace, které definuje Material design, a umožnit lepší manipulaci s oblíbenými položkami.

Stav aplikace se zejména odvíjí od složitosti protokolu BACnet®, který bylo třeba důkladně prostudovat za účelem pochopení fungování systémů HVAC. Využití knihovny, vzhledem k nedostatečné dokumentaci, také nebylo triviální, ale podařilo se úspěšně. Vzniklý systém je robustní a umožňuje snadné rozšíření o další objekty.

Další rozvoj

Jak bylo zmíněno výše, aplikace rozhodně není finální verzí pro uvolnění do produkce. Je třeba implementovat větší množství objektů a celý projekt předat do rukou společnosti, která má možnosti pro její využití jako součást zakázek na automatizaci budov. Mimo funkcí již zmíněných v této práci se ovšem naskýtá velké množství dalších funkcí, o které by tento systém mohl být obohacen.

Super-user verze

Aplikace BACdroid slouží běžným uživatelům, pro které je třeba vytvořit prostředí takové, aby mohli snadno ovládat komponenty HVAC systému, na které má takový uživatel v běžné praxi oprávnění. Princip konfiguračních souborů a mobilní zobrazení ovšem umožňuje i využití pro servisní techniky.

Jako scénář využití si lze například představit, že při výpadku určitého prvku HVAC systému bude mít servisní technik v místě regulátoru, nepřístupném veřejnosti, nalepený NFC štítek. Po jeho načtení se servisní aplikace napojí na lokální síť, načte konkrétní objekt a zobrazí veškeré informace o objektu.

Aplikace by musela být rozšířena o možnost získávat chybová a varovná hlášení z objektu, možnost zobrazit všechny vlastnosti daného objektu, a případně o další služby, které nejsou přístupné pro běžného uživatele. Servisní technik by tak měl možnost diagnostikovat problém, změnit hodnoty objektu a případně i restartovat BACnet® zařízení, pouze za pomoci mobilního telefonu. Běžnou praxí je využití osobního počítače nebo notebooku a dalších hardwarových nástrojů.

V této verzi by bylo možné odklonit se od současného návrhu rozhraní, vzhledem k tomu, že servisní technik má přehled o BACnet® zařízeních, objektech a vlastnostech. Záložka nástrojů pro discovery by byla zpřístupněna, spolu s dalšími funkcemi, jako například manuální volba priority příkazu.

Vzdálené ovládání

Pro využití v domácnostech i veřejných budovách by aplikace mohla být rozšířena o webovou část, která by zprostředkovávala požadavky zadané v mobilním zařízení a předávala tyto akce do lokální sítě pro zařízení BACnet®. To by například umožnilo zapnout topení před příjezdem domů, nebo odebírat hlášení o problémech, které by jinak byly dostupné pouze na místě lokální sítě BACnet®.

Konfigurační nástroje

Pro vytváření konfiguračních souborů by bylo vhodné vytvořit desktopový nástroj, který by umožnil správci BACnet® systému vybrat určité objekty a jejich vlastnosti a z nich vytvořit konfigurační soubory pro různá datová média. Například zadání velikosti NFC štítku by umožnilo vytvořit konfiguraci co možná nejobsáhlejší, ale zároveň takové, která by se do štítku stále vešla.

Knihovna BACnet4J je využitelná nejen pro platformu Android, ale i pro desktopovou verzi jazyka Java. Zkušenosti získané vývojem mobilní aplikace BACdroid mohou být při vytváření takového konfiguračního nástroje velice přínosné.

Literatura

[1] ASHRAE: A Data Communication Protocol for Building Automation and Control Networks. 2012, ANSI/ASHRAE Standard 135-2012.

[2] Google: Fragments.
URL http://developer.android.com/guide/components/fragments.html

[3] Google: Material design.
URL https://www.google.com/design/spec/material-design/introduction.html

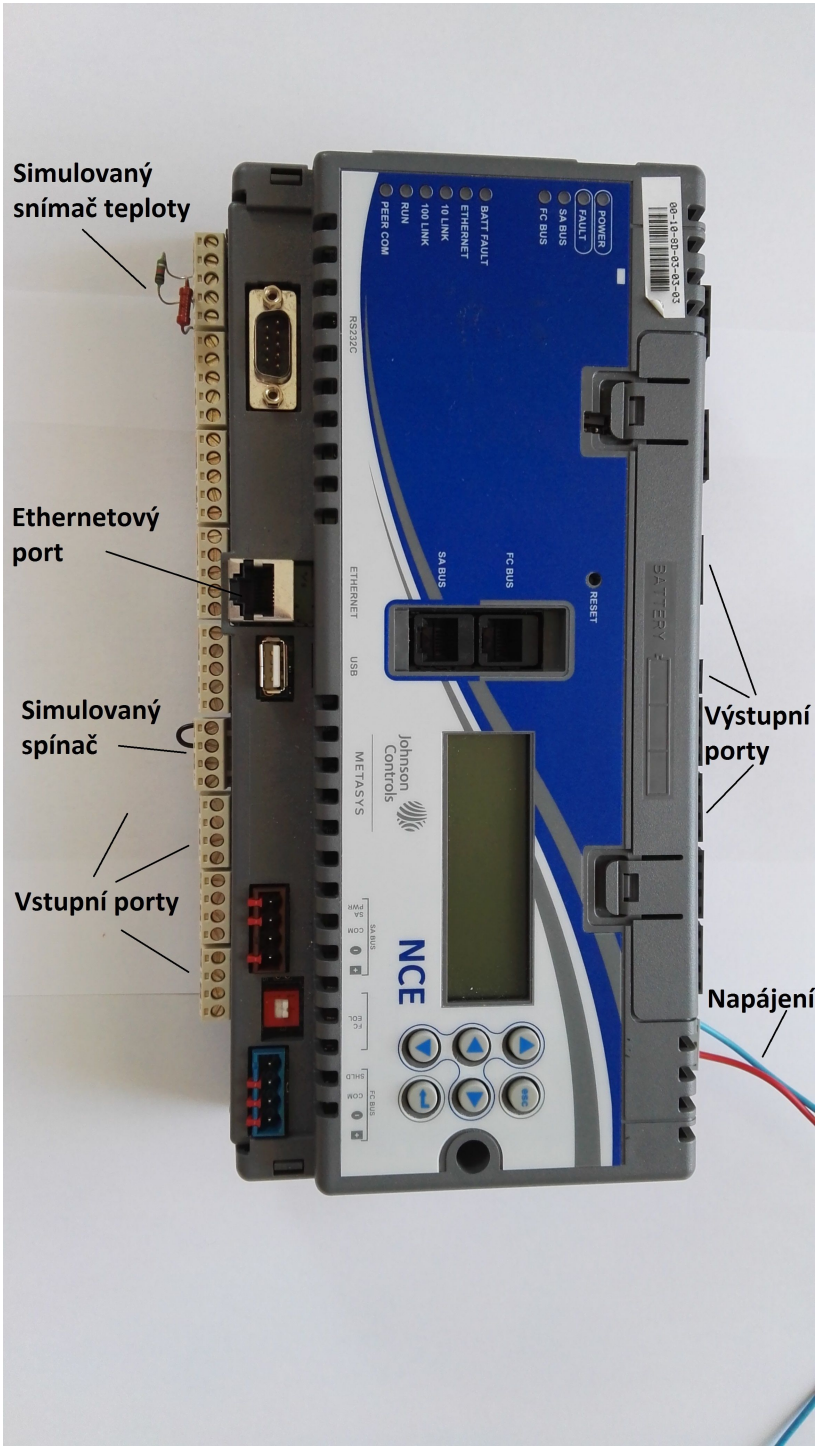
[4] Google: Async Task. 2016.
URL http://developer.android.com/reference/android/os/AsyncTask.html

[5] Google: Handler. 2016.
URL http://developer.android.com/reference/android/os/Handler.html

Příloha **A**

Ukázka BACnet® zařízení

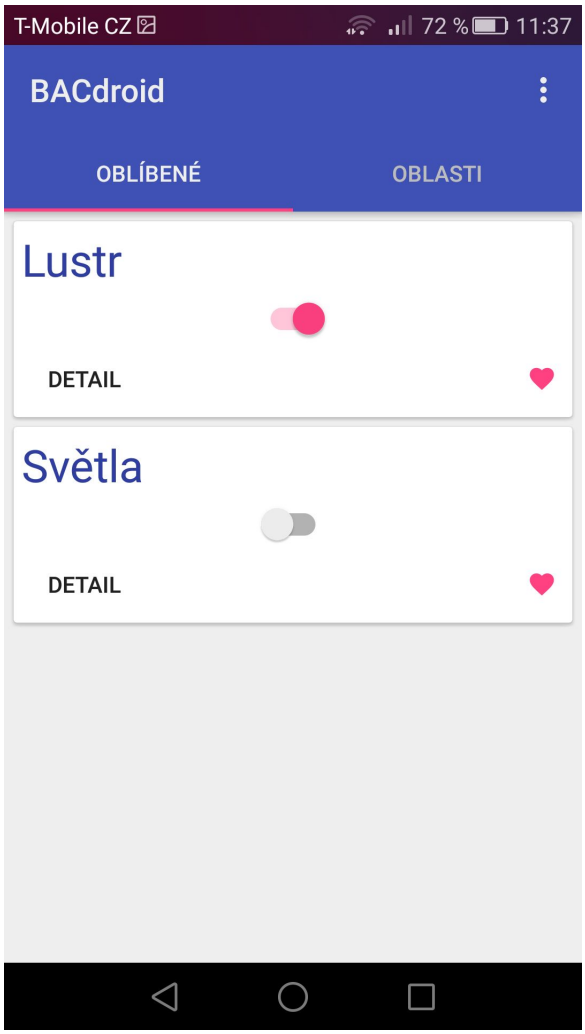
Při vypracování práce byl k dispozici regulátor NCE, který je na obrázku níže. Regulátor byl připojen Ethernetovým kabelem k WiFi routeru a do různých portů byla připojena periferní zařízení nebo jejich simulace. Například propojením svorek binárního vstupu vodičem lze simulovat sepnutý spínač, překlenutím svorek analogového vstupu rezistorem lze simulovat snímač teploty.



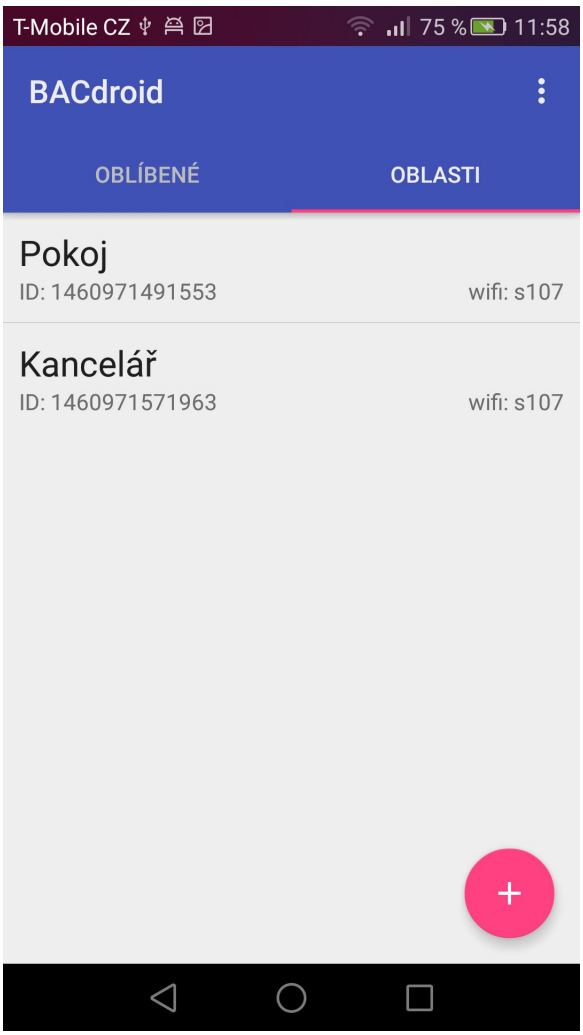
Obrázek A.1: Zařízení NCE

Příloha B

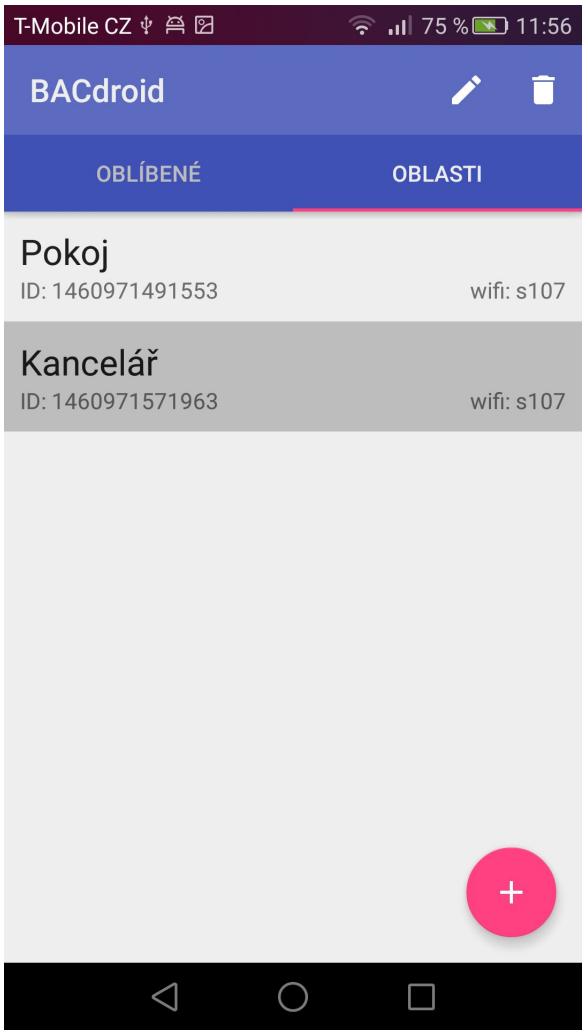
Obrazovky aplikace



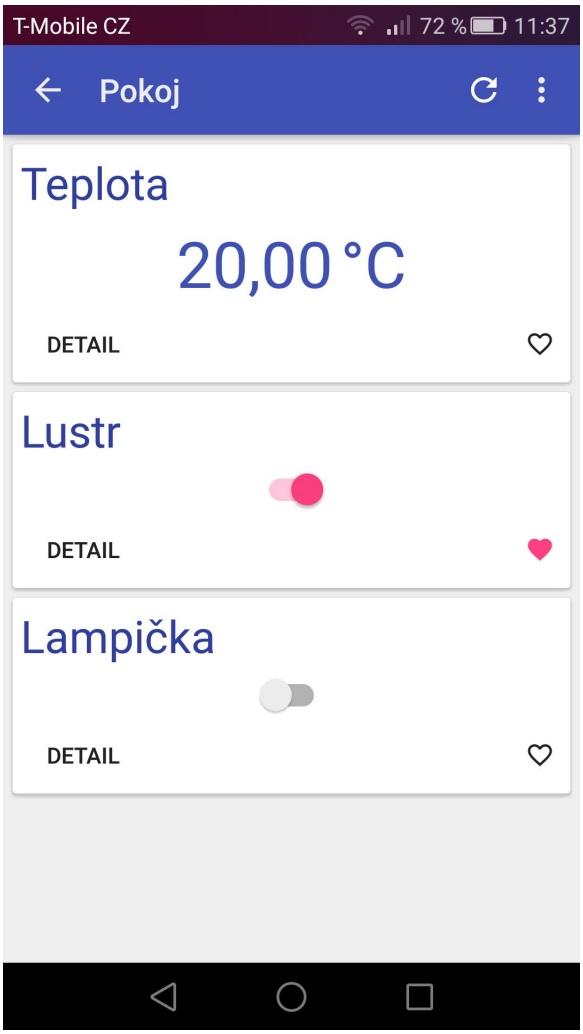
Obrázek B.1: Záložka oblíbených položek



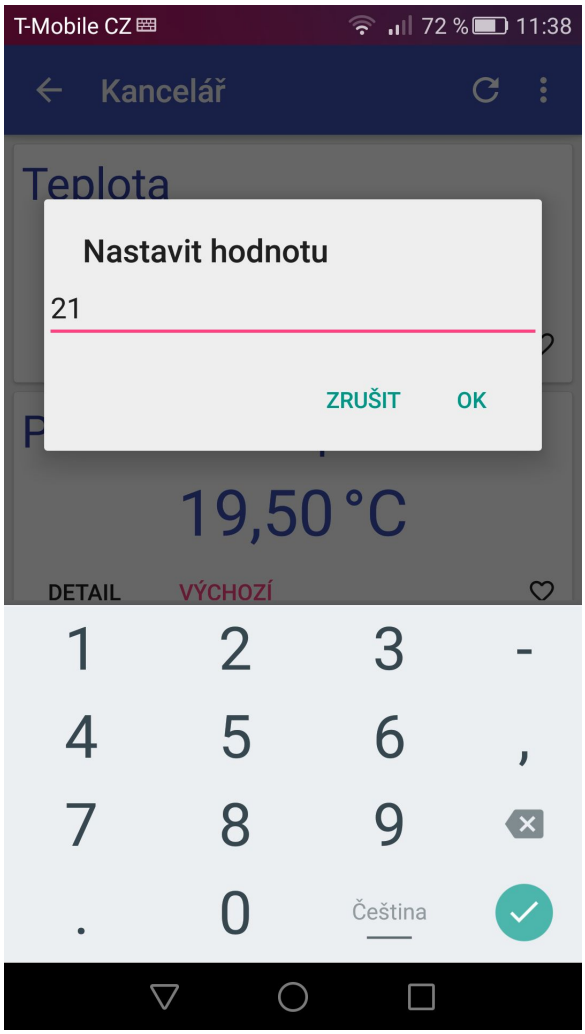
Obrázek B.2: Záložka oblastí



Obrázek B.3: Úprava oblasti



Obrázek B.4: Zobrazení objektů oblasti



Obrázek B.5: Zadání hodnoty

Příloha C

Ukázka konfiguračního souboru

Listing C.1: Test

```
<area_config name="Ukazkova laborator" id="1460971602108"
    version="1.0">
    <network ssid="BACnet-testing" type="wifi"/>

    <object template="1" viewable_id="1">
        <bacnet_object device_id="505085" object_id="3000074"
            object_type="2">
            <property commandable="true" type="85"
                view_id_name="text_present_value">
                <handler format="%.2f" type="2"/>
            </property>
            <property override="Test analog value"
                override_permanent="true" type="77"
                    view_id_name="text_header">
                <handler type="1"/>
            </property>
            <property override="%" override_permanent="true"
                type="117" view_id_name="text_units">
                <handler type="1"/>
            </property>
            <property type="69" view_id_name="text_min_value">
                <handler format="%.2f" type="2"/>
            </property>
            <property type="65" view_id_name="text_max_value">
                <handler format="%.2f" type="2"/>
            </property>
        </bacnet_object>
    </object>

    <object template="1" viewable_id="2">
        <bacnet_object device_id="505085" object_id="3000078"
            object_type="4">
            <property commandable="true" type="85"
                view_id_name="switch_present_value">
                <handler type="3"/>
```

```
29              </property>
30              <property override="Test binary output"
                    override_permanent="true" type="77"
31                      view_id_name="text_header">
32                  <handler type="1"/>
33              </property>
34          </bacnet_object>
35      </object>
36
37      <object template="1" viewable_id="3">
38          <bacnet_object device_id="505085" object_type="19"
                object_id="3000094">
39              <property type="77" view_id_name="text_header"
                    override="Rezim" override_permanent="true">
40                  <handler type="1"/>
41              </property>
42              <property type="85" view_id_name="text_present_value"
                    commandable="true">
43                  <handler type="4"/>
44              </property>
45              <property type="110" view_id_name="array_state_texts"
                    >
46                  <handler type="5"/>
47              </property>
48          </bacnet_object>
49      </object>
50
51      <object template="1" viewable_id="4">
52          <bacnet_object device_id="505085" object_type="19"
                object_id="3000095">
53              <property type="77" view_id_name="text_header"
                    override="Rezim - vice stavu" override_permanent="
                    true">
54                  <handler type="1"/>
55              </property>
56              <property type="85" view_id_name="text_present_value"
                    commandable="true">
57                  <handler type="4"/>
58              </property>
59              <property type="110" view_id_name="array_state_texts"
                    >
60                  <handler type="5"/>
61              </property>
62          </bacnet_object>
63      </object>
64
65      <object template="1" viewable_id="5">
66          <bacnet_object device_id="1004" object_id="2132"
                object_type="4">
67              <property commandable="true" type="85"
                    view_id_name="switch_present_value">
68                  <handler type="3"/>
69              </property>
```

```
70              <property override="Test binary output"
                    override_permanent="true" type="77"
71                      view_id_name="text_header">
72                  <handler type="1"/>
73              </property>
74          </bacnet_object>
75      </object>
76
77 </area_config>
```

Příloha D

Seznam použitých zkratek

BACnet Data Communication Protocol for Building Automation and Control Networks

MS/TP Master-Slave/Token Passing

UI User Interface

XML Extensible Markup Language

MAC Media Access Control

ASHRAE American Society of Heating, Refrigerating and Air-Conditioning Engineers

ISO International Organization for Standardization

UDP User Datagram Protocol

PICS Protocol Implementation Conformance Statement

BIBBs BACnet Interoperability Building Blocks

HVAC Heating, Ventilation and Air Conditioning

JSON JavaScript Object Notation

USB Universal Serial Bus

EIA Electronic Industries Alliance

Příloha **E**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `bacdroid.apk` ..... instalační balíček implementované aplikace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `diplomova prace.pdf` ..... text práce ve formátu PDF
  - `zadani.pdf` ..... zadání práce ve formátu PDF