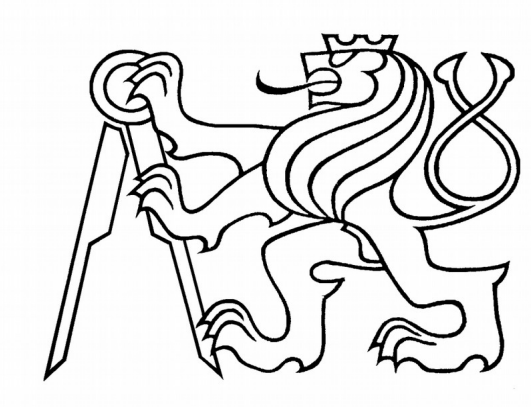
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA DOPRAVNÍ



Adam Eichler

MODEL KOMUNIKACE VOZIDLOVÉHO INFORMAČNÍHO SYSTÉMU PRO CESTUJÍCÍ

Bakalářská práce

2019

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K620..... Ústav dopravní telematiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Adam Eichler

Kód studijního programu a studijní obor studenta:

B 3710 – ITS – Inteligentní dopravní systémy

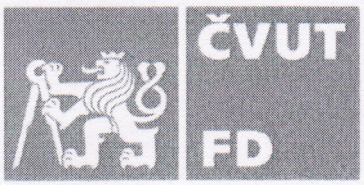
Název tématu (česky): **Model komunikace vozidlového informačního systému pro cestující**

Název tématu (anglicky): Communication Model of Vehicle On-board Information System for Passengers

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte osnovou uvedenou v následujících bodech:

- analyzujte současné možnosti propojení komponent vozidlových informačních systémů
- vyberte rozhraní vhodné pro realizaci modelu v laboratoři za použití dostupného HW a SW
- navrhněte a realizujte model části VIS využívajícího dané rozhraní (alespoň 2 prvky systému)
- vyhodnoťte vhodnost modelu pro účely výuky a flexibilitu pro účely rozšiřování o další prvky



Rozsah grafických prací: dle požadavků vedoucího práce

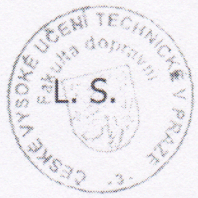
Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: vozidlové informační systémy v dopravě
komunikační sítě a protokoly
programování v jazyce C++
databázové systémy

Vedoucí bakalářské práce: **Ing. Milan Sliacky**

Datum zadání bakalářské práce: **7. června 2017**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **26. srpna 2019**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia



Ing. Zuzana Bělinová, Ph.D.
vedoucí
Ústavu dopravní telematiky

doc. Ing. Pavel Hrubeš, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Adam Eichler
jméno a podpis studenta

V Praze dne 30. listopadu 2018

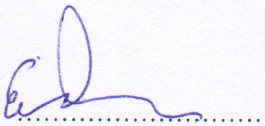
Poděkování

Tímto bych chtěl poděkovat hlavně svému vedoucímu Ing. Milanu Sliackému, Ph.D. za dlouholetou spolupráci v oblasti informačních a odbavovacích systému ve veřejné dopravě spočívající v rovnocenné výměně odborných informací a možnosti se aktivně zapojit v činnosti Zkušební laboratoře odbavovacích a informačních systémů na Fakultě dopravní ČVUT. Dále bych chtěl poděkovat svému současnému nadřízenému Ing. Janu Šimůnkovi za projevenou důvěru při zavádění teoreticky nabitých znalostí do praxe Pražské integrované dopravy na Oddělení technického rozvoje Regionálního organizátora pražské integrované dopravy. (ROPID)

Čestné prohlášení

Prohlašuji, že jsem tuto písemnou studii bakalářské práce vypracoval samostatně, pouze za odborného vedení vedoucího práce Ing. Milana Sliackého, Ph.D. Dále prohlašuji, že veškeré podklady a zdroje, ze kterých jsem čerpal, jsou uvedeny v seznamu použité literatury v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací. Nemám závažný důvod proti užívání tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 25. 8. 2019



Adam Eichler

Abstrakt

Autor: Adam Eichler

Název bakalářské práce: Model komunikace vozidlového informačního systému pro cestující

Škola: České vysoké učení technické v Praze, Fakulta dopravní

Rok vydání: Praha 2019

Počet stran: 48

Cílem této práce je porovnání současných možností komunikace mezi prvky informačního systému pro cestující ve vozidle MHD. Na základě výběru podle kritérií modularity a využitelnosti při výuce je zvolen jeden způsob komunikace pro tvorbu funkčního modelu.

Klíčová slova: OBU, OIS, MHD, VDV301, IBIS, IBIS-IP, HTTP, mDNS, DNS-SD, Qt, C++, ROPID, LCD, ethernet

Abstract

Author : Adam Eichler

Name of bachelor thesis : Communication Model of Vehicle On-board Information System for Passengers

School: České vysoké učení technické v Praze, Fakulta dopravní

Year of Publication : Prague 2019

Pages: 48

The aim of this paper is to analyse contemporary possibilities of device-to-device communication between components of vehicle information system in public transport. Based on this brief research, one method is wisely chosen based on modularity and usability for education to create a model.

Keywords: OBU, Passenger information system, MHD, VDV301, IBIS, IBIS-IP, HTTP, mDNS, DNS-SD, Qt, C++, ROPID, ethernet

Obsah

1 Úvod

1.1 Předmluva

Odbavovací a informační systémy ve vozidlech hromadné dopravy pokrývají široké odvětví dopravní telematiky. Obsahuje jak prvky pro informování cestujících (vnější maticové zobrazovače, hlásiče zastávek), tak zařízení pro správné účtování poplatku za využití dopravního prostředku (odbavovací systém). Mezi okrajové oblasti, které OIS pokrývá, je také dopravní telemetrie. Systém je často schopen měřit údaje jako teplota ve vozidle nebo aktuální rychlost vozidla a sdružovat je na centrálním úložišti pro pozdější zpracování.

Dříve vládly celému odvětví zařízení navrhované pomocí zastaralých součástek, ovšem s téměř neuvěřitelnou životností. Dodnes jsou hojně v provozu zařízení stará i 20 let. Ovšem jako již nestačí jejich paměť (v řádech MB), nestačí už jejich procesorový výkon na nové nároky na informování cestujících v reálném čase.

Jejich roli převzaly průmyslové počítače konvenční architektury. Pro potřeby výroby odpadla potřeba vývojářů specializujících se na mikrokontroléry, ustupují průmyslové sběrnice a vše se zjednodušuje. Komunikace mezi komponenty se pak stává pouze záležitostí protokolu. Při současné situaci dodavatelé odbavovacích systémů tvoří s dodavateli informačních systémů výhodná spojenectví, která se snaží kontrolovat poskytováním/neposkytováním informací, výrobní dokumentace, nebo například už zmíněných proprietárních komunikačních protokolů.

Díky tajnostem mezi dodavateli nemá organizátor IDS možnost zasahovat do způsobu přenášení informací ani do rozsahu. Pokud ano, musí změnu vyžadovat přes dopravce, což neúměrně komplikuje celý postup. Zároveň dodavatelé nevěnují dostatečné množství času přímé komunikaci s organizátorem, neboť s ním nejsou nijak finančně vázaní.

Proto vznikla iniciativa CDV a ROPID zvolit jednotný komunikační standard pro zařízení se zvýšenou hustotou informací, hlavně TFT-LCD vnitřní zobrazovací panely s tím, že pro vnější maticové panely dočasně bude využíván IBIS, výhledově nově zvolený standard pro ovládání TFT-LCD.

1.2 Cíl práce

Účelem této práce je vybrat vhodný způsob komunikace mezi prvky informačního systému pro cestující v prostředku hromadné dopravy. Podobné systémy v současnosti pomocí zobrazení na zobrazovacích zařízeních (LCD monitory, LED maticové displeje) a akusticky pomocí reproduktorů poskytují cestujícímu informace o lince, trase a aktuální poloze na daném spoji. Zvolený způsob komunikace musí umožňovat tyto funkcionality, musí být rozšiřitelný a nesmí principem svého fungování tvořit úzké hrdlo komunikace, a to ani výhledově.

Pro model výsledného informačního systému byly zvoleny následující komponenty - palubní jednotka řidiče (OBU) a vnitřní informační panel na bázi zobrazovače TFT. Volba těchto komponent zároveň udává podmnožinu komunikačních protokolů, které budou implementovány v modelu.

2 Analytická část

2.1 Analýza současných komunikačních rozhraní

Současné komponenty vozidlových informačních systémů jsou v drtivé většině případů řízeny z jednoho centrálního prvku, který sdružuje data o jízdních řádech, sbírá telemetrická data o vozidle, provozuje vozovou část preference MHD, umožňuje ovládání řidičem a hlavně řídí procesy na základě daných vstupů. Tento centrální prvek má mnoho různých rozhraní a komunikačních sběrnic.

V průběhu vývoje těchto systémů postupně přibývaly prvky, jimž současné sběrnice nepostačovaly a proto postupně do zařízení přibývaly nové sběrnice. Různé typy pro různá zařízení od různých dodavatelů. V současnosti se však začíná projevovat nový trend, a to postupné převádění všech komponent na připojení po sběrnici Ethernet. Tím dochází ke sjednocování komunikace po HW stránce, ale komunikační protokoly jsou stále roztříštěné.

2.1.1 RS485

Toto rozhraní je velmi volně pojato. Z fyzikálního hlediska dochází k přenášení dat pomocí měření rozdílu napětí mezi dvěma vodiči. Velikost napěťového rozdílu není pevně dána. Pro obousměrnou komunikaci je možné využít jak zdvojeného vedení (duplex) nebo komunikace po jednom páru vodičů s řízením směru datového toku (half-duplex). Toto rozhraní nemá v dopravě žádný jednotný protokol. Několik výrobců má svůj protokol zaveden:

- Mobitec
- BUSE
- EM-TEST

Mobitec Přesto, že neexistuje žádná veřejně dostupná oficiální dokumentace pro komunikaci s panely Mobitec, přes RS485, minimálně dvěma programátorům se pomocí reverzního inženýrství podařilo komunikační protokol implementovat pro použití s programovacími jazyky **C** a **Python**. `https://github.com/bjarnekvae/pymobitec-flipdot` `https://github.com/duffrohde/mobitec-rs485`

Protokol EM-TEST V systémech dodaných firmou EM-TEST propojuje RS485 části systému, které nevyžadují rychlou komunikaci. Využívá dvou vodičů, half-duplexní komunikaci a sběrnicové uspořádání. Hlavně propojuje starší informační panely. Buď přímo připojené na sběrnici RS485, nebo přes jednotku EM RDi, jejíž vstupem je sběrnice RS485 a výstupem je pomalejší sběrnice IBIS (detailněji zmíněná dále). Protokol komunikace je neveřejný. Pro komunikaci s odbavovacími zařízeními EM-TEST jej v současnosti využívají firmy Herman a Bustec ve svých zobrazovacích panelech. Pro zobrazení na LED panelech má protokol podporu pro dva módy:

Obrázek 1: Grafický mód zobrazovacího panelu EmTest(Autor, 2017)

Grafický mód Grafický mód umožňuje odeslat do panelu přesnou konfiguraci LED diod pro jednotlivé zobrazení (bitmapa). Backoffice pro palubní PC nabízí integrovaný editor bitmap jenž umožňuje složit výslednou bitmapu z manuálně vytvořených obrazců rozsvěcením jednotlivých diod případně konverzí běžných počítačových fontů do diodového rastru.

Databáze obrazců je trvale uložena v palubním počítači a do zob. panelu se posílají jen jednotlivé obrazce až v okamžiku použití. Tedy updaty databáze není nutné provádět pro každý panel zvlášť, ale stačí celou databázi nahrát jen do palubního počítače.

Obrázek 2: textový mód zobrazovacího panelu EmTest (Daniel Kelnar, 2015)

Textový mód Textové řízení probíhá pomocí odesílání samotných textů doplněných o velikosti fontů a mód zobrazení (běžící text, stojící text). Změnit písmo zobrazení textů není jednoduše možné, protože jsou písma pevně integrována do programu panelu.

Krátký odposlech komunikace odbavovacího zařízení se zobrazovacím panelem v textovém módu. Jde o binární formát bez zjevné logiky. Je pravděpodobné, že při získání dostatečného vzorku odposlechových dat ze sběrnice a odpovídajících zobrazení na panelu by bylo možné protokol prolomit. Záznam je pořízen pomocí převodníku USB→RS485 a volně dostupného programu Termite.

```
a4 ae 04 00 00 87 2c 40  18 4e 04 09 02 90 00 5a  |...S..,@.N.....Z|
6c a1 6e 2c 2c 61 75 74  2e 6e a0 64 72 2e 00 e0  |l.n,,aut.n.dr...|
53 60 03 53 a4 ae 04 53  00 87 2c 40 18 4e 04 09  |S`.S...S..,@.N..|
02 90 00 5a 6c a1 6e 2c  2c 61 75 74 2e 6e a0 64  |...Zl.n,,aut.n.d|
```

```
72 2e 00 e0 53 60 03 53  a4 ae 04 53 00 87 2c 40  |r...S`.S...S..,@|
92 52 02 13 02 90 00 38  32 30 37 34 31 20 2d 20  |.R.....820741 - |
5a 6c a1 6e 2c 2c 61 75  74 2e 6e a0 64 72 2e 20  |Zl.n,,aut.n.dr. |
2d 20 48 76 6f 7a 64 6e  a0 2c 20 44 6f 6c 6e a1  |- Hvozdn., Doln.|
20 7a 61 73 20 2d 20 48  76 6f 7a 64 6e a0 2c 20  | zas - Hvozdn., |
4f 73 6d 65 6b 20 2d 20  5a 6c a1 6e 2c 9b 74 a1  |Osmek - Zl.n,.t.|
70 61 2c 70 6f 91 2e 7a  20 2d 20 5a 6c a1 6e 2c  |pa,po..z - Zl.n,|
9b 74 a1 70 61 2c a8 6b  6f 6c 61 20 2d 20 5a 6c  |.t.pa,.kola - Zl|
a1 6e 2c 2c 56 79 73 6f  6b a0 20 6d 65 7a 20 2d  |.n,,Vysok. mez -|
20 5a 6c a1 6e 2c 2c 61  75 74 2e 6e a0 64 72 2e  | Zl.n,,aut.n.dr.|
00 ff 15 60 03 53 a4 ae  04 53 00 87 2c 40 92 52  |...`.S...S..,@.R|
```

2.1.2 IBIS (VDV300)

Toto rozhraní je jedno z mála obecně rozšířených standardů v průmyslu informačních a odbavovacích systémů. Vzniklo jako produkt německého sdružení autobusových dopravců VDV (Verband Deutscher Verkehrsunternehmen) a mělo vytvořit jednotné rozhraní pro komunikaci všech prvků informačního systému ve vozidle. Tento standard není dostupný zdarma, dokumentaci je stále nutno zakoupit. Topologicky se jedná o sběrnici využívající 4 vodiče.

- **SD** - hlavní odesílací linka
- **MS** zpětná odesílací linka
- **ED** hlavní linka přijímací
- **ME** zpětná linka přijímací

Napěťové úrovně jsou 24 V a 0 V. Komunikace je sériová - baudrate je 1200 bps, parita sudá, 7 databitů a 2 stop bity. Využívá master-slave architekturu. Master zařízení (OBU) vysílá příkazy po hlavní lince a jednotlivě zároveň vysílá dotazy na stav zařízení. Po přijímací lince tedy vždy komunikuje jen dotázané zařízení. Za tímto účelem mají zařízení stejného typu vlastní adresu, aby nedocházelo k odpovědím z více zařízení naráz. Připojení více master zařízení na sběrnici není přípustné a vede k destrukci obou připojených zařízení.

Zpráva vyslaná z master zařízení vždy obsahuje kombinací čísel a písmen pro identifikaci typu zprávy, případně konkrétního adresáta zprávy. Tento identifikátor je následován mezerou a obsahem samostatné zprávy a je zakončen koncovým znakem a kontrolním součtem.

Kategorie zpráv z 973 - vyvolání cíle 973 z databáze uložené ve všech připojených panelech

u 1231 nastavení času 12:31 na všech zařízeních (zobrazovač času a pásma, označovače)

Původní německá norma ovšem pro provoz v České republice nedostačovala kvůli absenci české znakové sady a kvůli nemožnosti text formátovat pomocí textového řízení. Proto čeští výrobci upravili normu do podoby obecně nazývané IPIS. Tato norma už

obsahuje nástroje pro formátování textu na zobrazovacích panelech (velikost fontu, mezery, poloha zobrazení na panelu) a českou znakovou sadu. Mezi jednotlivými výrobci se tyto standardy liší, například znakovou sadou, (Kamenických vs. KOI8-CS).

Tento standard se stále využívá pro připojení zobrazovacích panelů a v nabídce u svých prvků IS jej mají všichni čeští výrobci. Nevýhodou je specifická kabeláž $2 \times 2 \times 0,5$ s barvami hnědá, žlutá, bílá a zelená. Vzhledem k velkému množství mědi, které obsahuje, je drahá a není běžně k dostání. Kapacity pro další rozšiřování tohoto standardu jsou již vyčerpány, minimálně kvůli nízkému datovému toku.

Rozbočovače kabeláže jsou plně pasivní. Neexistuje standardní definovaný konektor, periferie jsou zpravidla vybaveny svorkovnicemi. převodník lze udělat pasivní HW redukcí s převodníkem úrovní,

Rozšíření

Nevýhody roztříštěnost standardu, omezená délka zprávy, dlouhá odezva pro přijímač nevidomého

Záznam příkazů odesílaných zařízením Emtest EM126i přes komunikační jednotku EMRDi, pořízený pomocí převodníku IBIS→RS232→USB a volně dostupného programu IBIS Utility. `http://cybox.ib-luehning.de/ibis/`

```
in->  yS<CR>
in->  aA1 <0x1b>b<LF><0x1b>zt<0x1b>W<0x1b>0 <0x0c><0x1b>R<0x1b>!780600
    Ludm<0x0e>Irov-La<0x0e>Skov<0x0c><0x1b>:<0x0e>celechovice  n.H.-
   Prost<0x0e>Ejov<LF><0x1b>zt<0x1b>W<0x1b>0 <0x0c><0x1b>zt<0x1b>W<0
   x1b>0 <0x0c><0x1b>U<0x1b>!Plumlovsk<0x0e>A OC<0x0c><0x1b>c<0x0b><
   CR> //příkaz pro přední panel
in->  aA:2600 23.1.2017 13:56 <CR> // příkaz pro vnitřní panel
in->  yS<CR>
in->  aA2 <0x1b>c<0x1b>l<0x12><0x1b>U<0x1b>!Plumlovsk<0x0e>A OC<0x0c
   ><0x1b>c<0x0b><CR>
in->  eA0122<CR>
in->  yS<CR> //dotaz na stav označovače
in->  eA0122<CR> //neznámý příkaz
in->  yS<CR>
in->  d23017<CR> //datum pro všechna zařízení
in->  yS<CR>
in->  aA:2Ludm<0x0e>Irov,Osp<0x0e>Elov <CR> // druhý obraz pro vnitřní
    panel
in->  dA17<CR>
in->  yS<CR>
in->  u1357<CR> // čas
```

2.2 Ethernet

Technologie ethernet v metalické variantě je známá hlavně z přístupových sítí do internetu. Lze ji ale využívat pro komunikaci několika zařízení v malé lokální síti. Pro propojení využívá kabely s různým množstvím kroucených párů vodičů. Na rozdíl od předchozích rozhraní se k ethernetu vážou konkrétní konektory, a to RJ45 s definovaným zapojením. K propojení většího množství zařízení je nutné použít aktivní rozbočovač: hub, switch nebo router. Pro dvě zařízení stačí využít kříženého kabelu. Stejně

jako RS485, ani Ethernet (zatím) nevyužívá v dopravních aplikacích žádný společný protokol.

Rychlosti přenosu dat určuje vždy nejpomalejší zařízení. Při použití gigabitových switchů a odpovídajících síťových karet je prakticky nemožné vyčerpat kapacitu spojení činností informačního systému. Vzhledem k univerzálnosti sběrnice je při správné konfiguraci IP adres možné provozovat několik různých systémů na stejné síti nezávisle na sobě. Výhodou je dostupnost hotových zařízení v *industrial grade* kvalitě, tím padem není nutné řešit výrobu specifického HW a dodavatelé se mohou plně soustředit na tvorbu SW. Zdroje ze síťových knížek Některé komunikační protokoly

2.2.1 IBIS over IP

Jedná se o komunikační protokol firmy Mikroelektronika. Využívá standardních příkazů normy IPIS a tyto balí do datových paketů. Tento protokol je zatím jedinou implementací firmy Mikroelektronika pro komunikaci mezi panely a odb. zař. přes Ethernet. Pouze experimentálně pro Laboratoř OIS. Nasazení panelů komunikujících s odbavovacím zařízením Mikroelektronika pro Ethernetu není dodnes známo. Ve všech současných instalacích komunikace probíhá přes IBIS.

2.2.2 EPISNET

„Společnost Ing. Ivo Herman, CSc. vyvinula komunikační protokol pro ovládání LCD pro cestující - EPISNET, který na základě požadavků od různých provozovatelů v ČR sjednocuje jejich ovládání (dnes je již využíván DPO a.s., DSZO a.s., připravuje se DPMB a.s., Kordis a.s.)“.[1]

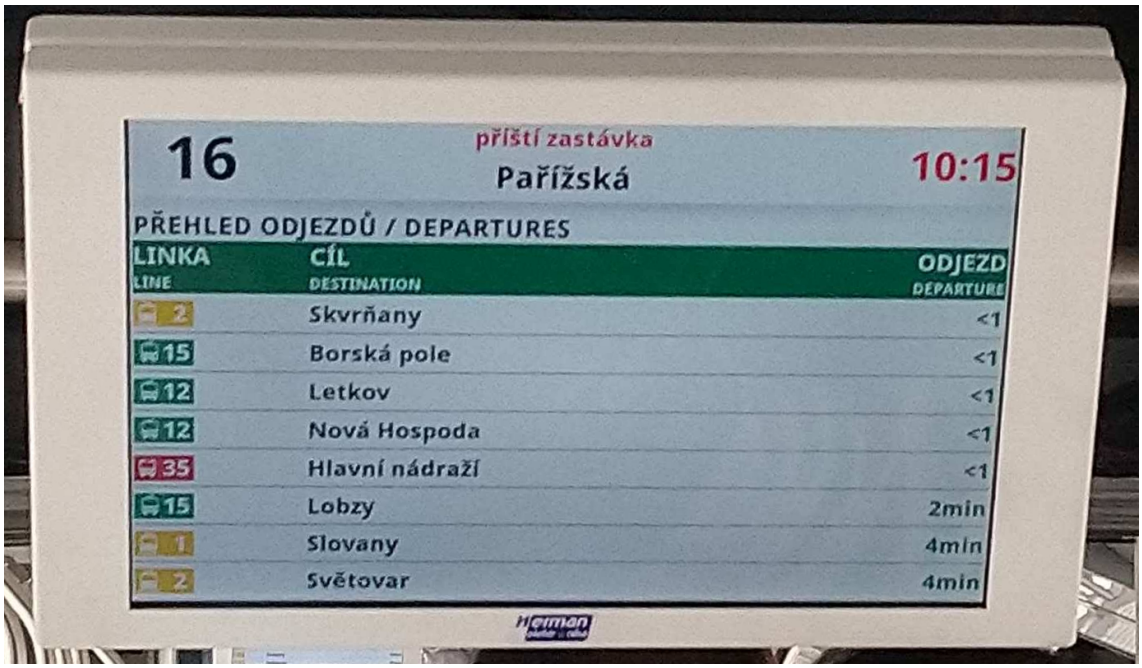
Dle tvrzení oficiálních stránek výrobce „Protokol bude zveřejněn na našich stránkách v krátké době“[1], by se mělo jednat o další veřejně dostupný komunikační protokol pro pokročilé ovládání vozidlových LCD, včetně specifických úprav pro Českou republiku. Přes veškeré ambice a úspěšné nasazení v provozu (např. zobrazování přestupů u PMDP) protokol zatím nebyl uveřejněn (25. 8. 2019) a proto není dál při výběru uvažován.

2.3 VDV301 (IBIS-IP)

Nově vznikající norma s cílem pokrýt komunikací všech prvků informačního systému ve vozidle, čímž navazuje na normu IBIS. Tato možnost komunikace je popsána nejdetailněji, neboť je zvolena jako nejvhodnější pro účely této práce, hlavně z důvodu rozšiřitelnosti a také proto, že tato norma je veřejně dostupná na síti Internet.

2.3.1 HW zapojení

Hlavní princip VDV301 spočívá v propojení komponent pomocí počítačové sítě. Ovšem již nespecifikuje, zda musí být drátová, či bezdrátová. Z hlediska bezpečnosti se počítá spíše s drátovým propojením metalickými vodiči s koncovkami RJ45. Volba kategorie kabelu leží pouze na dodavateli systému. S ohledem na umístění kabelů do okolí mnoha



Obrázek 3: LCD společnosti Herman komunikující přes EPISNET (Autor, 2019)

jiných vodičů pohonu vozidla a pomocných systémů vozidla je vhodné využít stíněný kabel, ať už FTP či STP.

Tato skutečnost otevírá nové možnosti ohledně volby komponent systému. Na rozdíl od kabelů pro sběrnici IBIS jsou sítové kabely levnější, dostupné v obchodech s počítači i elektronikou a není nutné je nakupovat od dodavatelů náhradních dílů pro nákladní vozidla/autobusy. S rozšiřováním různých reklamních systémů a systémů bezplatného internetu pro cestující se již začaly rozšiřovat i switche a routery certifikované pro provoz ve vozidlech MHD.

Konfigurace IP adres v síti Norma umožňuje dva různé způsoby konfigurace adres. Jeden je tradiční, kdy je všem komponentům v síti předem manuálně přidělena adresa tak, aby nedocházelo k duplicitám (tento postup v současnosti využívá např. EmTest).

Druhý postup zakládá na principu „zero-configuration“, což znamená, že lze připojovat nové komponenty bez potřeby dodatečné manuální konfigurace. To zaručuje hlavně protokol/služba DNS-SD. Tato umožňuje sdílet zařízením informace o poskytovaných „službách“. To znamená, že je možné požádat o přehled zařízení v síti, jaké „služby“ poskytují a na jakém portu a umožňuje s nimi na základě jejich síťových jmen komunikovat bez potřeby znát jejich aktuální IP adresu. Tento protokol hojně využívá firma Apple pro svoji službu Bonjour. Také lze doinstalovat pro systém Windows. U operačního systému Linux je mDNS implementováno do programu Avahi.

2.3.2 Princip funkce

Celý protokol využívá běžné síťové technologie. Zařízení spolu komunikují pomocí HTTP požadavků a odpovědí. Využívá se zde metod GET a POST. Každé zařízení

má své "služby", což jsou části programů poskytující navenek rozhraní na konkrétní adrese a portu.

2.3.3 Příklady použití pro dvojici OBU – TFT

TFT pravidelně obvolává OBU TFT jednou za určený interval provede dotaz na TFT HTTP metodou GET. Na tento dotaz OBU odešle XML soubor s daty v těle odpovědi.

Tento postup má několik nedostatků. Při příliš krátkém intervalu dotazů dochází ke zbytečnému zatížení OBU. Při příliš dlouhém intervalu obnovy jsou informace zobrazovány s příliš velkým zpožděním. Tento problém není tak závažný při použití tohoto postupu například pro zjišťování stavu jednotlivých zařízení.

Architektura publisher - subscriber Norma VDV301 pokrývá navíc opačný postup. Tento už vyžaduje HTTP metodu POST a využívá takzvaných odběrů. Dochází zde k převrácení rolí server/klient.

TFT zařízení vyšle POST požadavek, v jehož těle žádá o přihlášení k odběru výsledků z OBU. OBU odpoví potvrzení k odběru a přidá si adresu TFT do seznamu odběratelů. Nyní se role server/klient prohodí. Při každé změně vyšle OBU POST metodou data k zobrazení na TFT a TFT v odpovědi potvrdí přijetí dat. Tento postup umožní odesílat data jen ve chvílích, kdy dochází ke změně.

Obsah požadavku/odpovědi jsou XML formátovaná data. Tvar těchto dat určují XSD soubory dostupné na GitHubu sdružení VDV.

2.3.4 Zahájení odběru

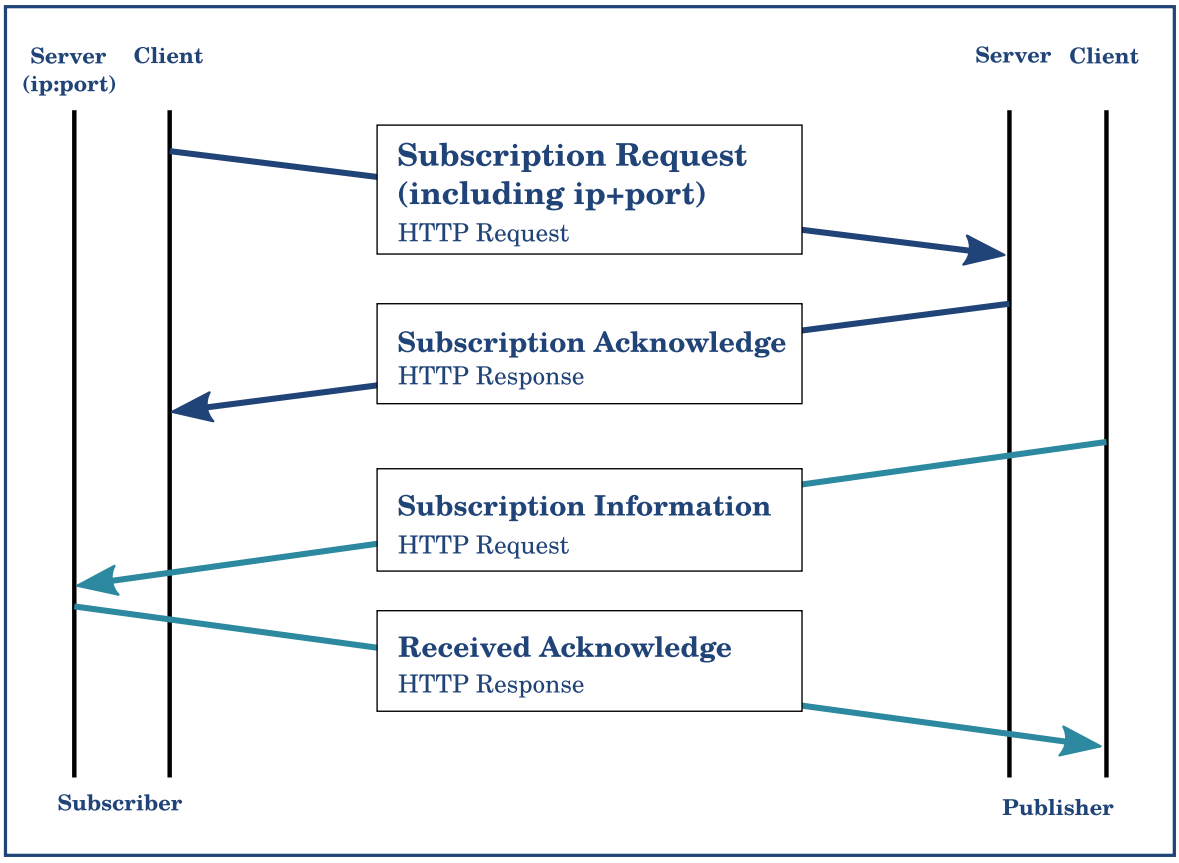
2.3.5 Záznam komunikace při odběru

Pořízeno programem Wireshark. Požadavek na odběr obsahující adresu a port, kam má palubní počítač zasílat data metodou POST.

```
POST /CustomerInformationService/SubscribeAllData HTTP/1.1
Content-Type: text/xml
Host: 192.168.1.100:8081
Content-Length: 284
Expect: 100-continue
Connection: Keep-Alive
```

```
HTTP/1.1 100 Continue
```

```
<?xml version="1.0" encoding="utf-8"?>
<SubscribeRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
   " xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <Client-IP-Address>
        <Value>192.168.1.128</Value>
    </Client-IP-Address>
    <ReplyPort>
        <Value>60011</Value>
    </ReplyPort>
</SubscribeRequest>
```



Obrázek 4: architektura Publisher - Subscriber (CDV, 2014)

```
HTTP/1.1 200 OK
```

Odpověď z OBU do TFT o úspěšném přihlášení k odběru.

```
Content-Length: 231
Content-Type: text/xml
Server: Microsoft-HTTPAPI/2.0
Date: Sun, 03 Mar 2019 18:46:48 GMT

<?xml version="1.0" encoding="utf-16"?>
<SubscribeResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Active>
    <Value>true</Value>
  </Active>
</SubscribeResponse>
```

2.3.6 Nejdůležitější služby

CustomerInformationService - běží na OBU. Zpracovává data o aktuální trase společně se stavem vozidla, jako je otevření dveří, případně aktivace tlačítka "STOP".

Device mgmt service - běžní na všech zařízeních.

2.3.7 Popis

- StopService
- StartService
- GetDeviceStatus
- GetServiceStatus?

Vzhledem k použití XML formátu dat je poměr značek ku samotnému obsahu poměrně nepříznivý, ale na druhou stranu je celá datová zpráva jednoduše zpracovatelná, neboť existuje mnoho knihoven pro práci s tímto formátem. Oproti formátu *json* je formát čitelný nejen storjově, ale jednoduše i pro člověka. Tento standard má do budoucna velký potenciál v možnosti vytváření nových služeb nebo nových tagů do XML souborů.

V současné podobě je standard uzpůsobený hlavně pro použití na TFT panelech. Pro použití na LED maticových panelech by musel být standard doplněn o možnost formátování textu, aby se vyrovnal současným možnostem textového řízení na sběrnici IPIS. (formátovací protokol firmy BUSE)

2.3.8 Struktura normy

Hlavní popis je součástí PDF souborů. `https://www.vdv.de/ip-kom-oev.aspx`

`301-2-1-sds-v2-1-commonstructure-enums_2018.pdf` `301-2-1-sds-v2-1-commonstructure-en`
`2018.pdf`

Konkrétní tvar XML je uveřejněn na GitHub účtu VDV: `https://github.com/VDVde/VDV301`

2.3.9 Použité technologie

- HTTP
- Hypertext transfer protocol
- HTTP GET
- HTTP POST
- TCP/IP

TCP/IP Protokol obsahuje mechanismy pro kontrolu integrity přenášeného celku a vyžaduje větší součinnost obou účastníků komunikace. Pomocí těchto mechanismů lze zaručit, že pokud nedojde k doručení nějaké části dat (datového paketu), bude se odesílatel pokoušet o o odeslání dat tak dlouho, dokud nebudou data odeslána úspěšně, případně dokud nevyprší stanovený time-out.

UDP Slouží pro rychlé odesílání dat u nichž není nutné zaručit integritu přenášeného celku. Hodí se v případech, kdy dochází k odesílání malých objemů dat, která jsou na sobě vzájemně nezávislá a neúspěšné doručení části dat neznehodnotí zbytek celku. Interakce nutná ze strany příjemce je minimální.

SQL Selective query language Jedná se o jazyk sloužící pro práci s relačními databázemi. Tyto nástroje umožňují tvořit relace v databázích, vytvářet nové záznamy, vytvářet sestavy a provádět základní práci s daty jako počítání průměru hodnot, řazení hodnot atd.

```
SELECT a.stop_order, b.name, a.time  FROM lineroutestoptime a
LEFT JOIN  stop b ON b.id = a.stop_id
LEFT JOIN lineroute c ON a.lineroute_id=c.id
LEFT JOIN line d ON c.line_ID = d.ID
WHERE d.licencenumber=820741
AND c.route_id =12
AND a.stop_order>=12
ORDER BY a.stop_order ASC LIMIT 5
```

XML Extensive markup language Výhody XML jazyka: jednoduchá čitelnosti i pro lidi, Nevýhody: velké množství jalových dat nenesoucích informace Jednoduché parsování pomocí metody DOM Atributy, tagy, parametry?

```
<?xml version="1.0"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't␣forget␣me␣this␣weekend!</body>
</note>
```

`https://www.w3schools.com/xml/schema_howto.asp`

XSD `https://www.w3schools.com/xml/schema_howto.asp` W3C XML Schema language je soubor obecně popisující strukturu konkrétního XML souboru. Specifikuje hierarchii tagů

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace
   ="https://www.w3schools.com" xmlns="https://www.w3schools.com"
   elementFormDefault="qualified">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

2.4 Srovnání specifik českého provozu s již definovanou funkcionalitou

Následující fotografie ukazuje testovací TFT-LCD dodaný do Zkušební laboratoře OIS FD ČVUT firmou Bustec pro účely testování a výuky, v přesné konfiguraci pro Wuppertal, tzn. komunikuje pomocí VDV301 ve verzi 1.0. Ukazuje současné možnosti zobrazení přestupů bez zásadních zásahů do normy. Úprava spočívá pouze v tvorbě číselníku dopravních prostředků pro zobrazení piktogramů. Tento panel posloužil také pro účely ověření kompatibility modelového palubního počítače s reálným zařízením. Později na tomto panelu byla provedena úprava šablony pro shodu s grafickým manuálem ROPIDu.

Obrázek 5: Panel určený pro reálný provoz v německém Wuppertalu (Autor, 2019)

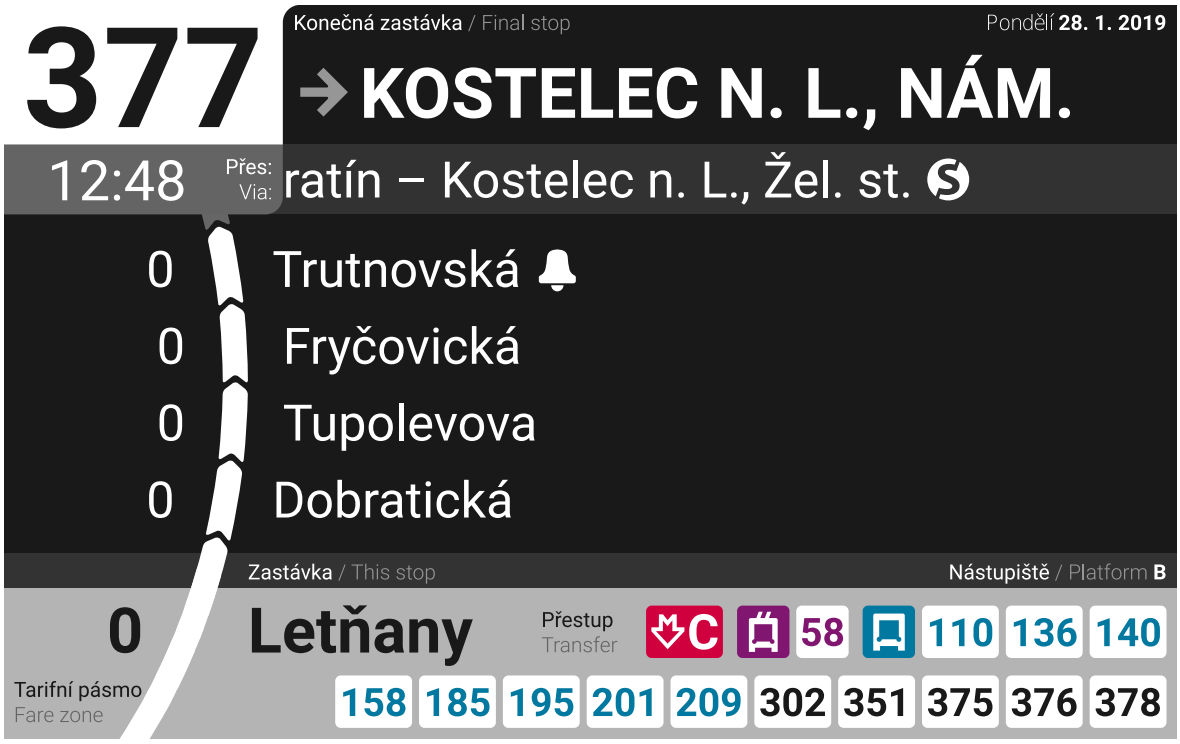
2.4.1 Iniciativa ROPID

Jedním z důvodů vzniku této bakalářské práce je iniciativa Regionálního organizátora pražské integrované dopravy (ROPID) a Centra dopravního výzkumu (CDV).

Vzhledem k minimální ochotě dodavatelů sdílet své proprietární komunikační protokoly je velmi složité řešit požadavky na informování cestujících v tak velkém systému, jako je Pražská integrovaná doprava (PID).

V současnosti jsou v PID provozovány informační systémy následujících dodavatelů:

- BUSE
  - IBIS (pal. PC Apex, JKZ, Mikroelektronika, Telmax)



Obrázek 6: LCD: vzorová domovská obrazovka [3]

Obrázek 7: LCD: vzorová stránka s přestupy[3]

  - Ethernet (pal. PC Telmax)

- Bustec
  - IBIS (pal. PC Mikroelektronika, Telmax)

  - Ethernet (pal. PC Apex, Telmax)
- Herman
  - Ethernet (pal. PC Konektel)
- JKZ
  - IBIS (pal. PC Mikroelektronika, Telmax)
  - Ethernet (pal. PC Telmax)
- Konektel
  - Ethernet (pal. PC Konektel)
- Novatronic
  - Ethernet (pal. PC Konektel)
- R&G Mielec
  - RS485 (pal. PC EmTest)
- Em-Test
  - IBIS (pal. PC APEX)

Tolik možných kombinací pal. PC (palubních počítačů) a zobrazovacích periferií vede k velice častým odchylkám reálného provozu zařízení od stavu stanoveného standardy kvality pro informování cestujících. Tyto odchylky je nutné řešit s každým dodavatelem/výrobcem/dopravcem zvlášť.

Částečně snížit složitost systému je možné pomocí sjednocení komunikačního protokolu. Obě organizace (ROPID a CDV) se shodly na volbě sběrnice Ethernet a komunikačního protokolu IBIS-IP (VDV301).

Specifické požadavky pro provoz PID jsou aktuálně definovány Standardy kvality (Odbavovací a informační zařízení ve vozidlech PID) `https://pid.cz/wp-content/uploads/2018/04/Priloha_10_OIS_ve_vozidlech_PID_vcetne_navaznych_priloh.pdf`

Tento dokument pokrývá specifikaci LCD panelů do vozidel po stránce obsahové a grafické. Samotný komunikační protokol je nutné určit a dospecifikovat na míru pro systém PID a to v maximální možné míře. Dopravci jsou organizátorem nuceni respektovat standardy kvality a používat pouze zařízení certifikovaná pro systém PID. Tímto vzniká nepřímý tlak na dodavatele standardy respektovat. Jakákoliv nejasnost/-možnost volby při implementaci komunikačního protokolu, která dodavateli vznikne, otevírá prostor pro nekompatibilitu různých částí celého informačního systému.

Přes rozsáhlou funkcionalitu definovanou v mateřské normě stále existují situace popsané grafickým manuálem, kdy oficiální verze VDV301 nedokáže přenést požadované informace popsané v grafickém manuálu. Naštěstí se většinu požadovaných dat podaří přenést pomocí značek <xxxRef></xxxRef>. Tyto značky byly definovány v první verzi normy, jsou často povinné, ale v současnosti není definován jejich obsah a jsou využívány pouze ve specifických provozech.Mezi tyto značky jsou dle návrhu společnosti Bustec vkládány vnořené xml prvky. Chybějící funkce jsou následující:

Statické přestupy Statické přestupky se od těch dynamických liší tím, že jde o pevně přiřazená čísla linek k zastávce, na které zastavují. Naopak dynamické přestupy jsou vázány na reálné on-line získané odjezdy s konkrétním časem. Norma zvládne odlišit garantovaný dynamický přestup od negarantovaného, ale statický už odlišit nedokáže. Proto je nutné tento příznak přidat do tagu ConnectionRef.

Zabarvení linek Dalším chybějícím článkem je podbarvení pole linky. Toto vzniká kombinací dopravního prostředku linky společně s typem linky (denní, noční, speciální, výluková). Tuto informaci je nutné přenášet už z palubního počítače, aby nebylo potřeba vytvářet a a aktualizovat číselník linek a jejich podbarvení přímo v panelu.

Příznaky zastávek Jednou z absurdit normy VDV301 je absence zastávek na znamení. Ve srovnání s ostatními požadavky ROPIDu se jedná o velmi zásadní funkci. Tradicí německých dopravních systému je režim, kdy jsou zastávky na znamení všechny a není je tedy nutné odlišovat zobrazením na panelu. Řešení využívá opět vnořeného XML, tentokrát do štítku StopRef.

Využívání vnořených XML souborů nenarušuje původní strukturu XML a tedy OBU dle specifikace ROPIDu by měla komunikovat s obecnými TFT-LCD nevyužívajícími *Ref* hodnoty.

3 Praktická část

Úkolem praktické části je vytvořit model informačního systému pro cestující pro potřebu výuky. Takový model by měl napodobovat reálný systém do té míry, že si jej studenti dokáží vizuálně i funkčně spojit reálným provozem.

Modelem se rozumí dva počítačové programy, jeden simulující palubní počítač (OBU) a druhý TFT-LCD displej.

3.1 Volba vývojového prostředí

Pro programování bylo nutné vybrat vhodné vývojové prostředí. Obecné požadavky:

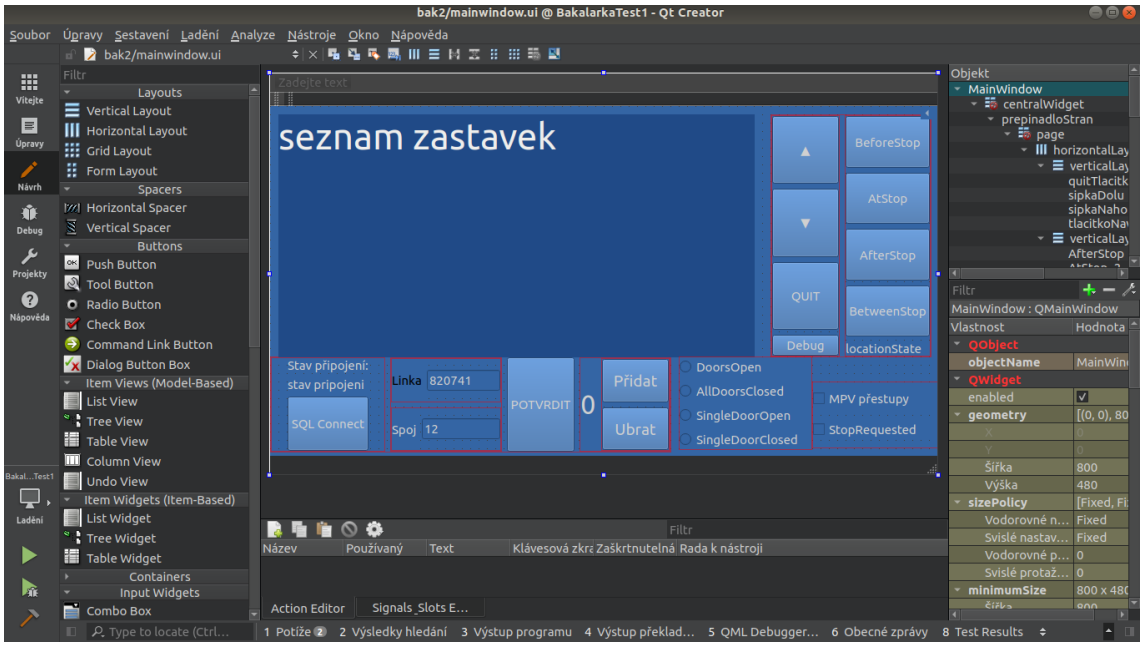
- Vývojové prostředí musí být multiplatformní
- Musí obsahovat nástroje pro tvorbu grafického rozhraní

Osobní požadavky:

- Programování v jazyce C++ Nebylo by v lidských silách se učit pro účely bakalářské práce kompletně nový programovací jazyk. Kvůli zkušenostem autora s jazyky PHP a C byl zvolen jazyk C++ a to kvůli analogické syntaxi s jazykem C a jednoduchou prací s datovými typy (například String), což by v jazyku C způsobovalo problémy.
- uživatelsky přívětivé IDE

Těmto požadavkům vyhovělo vývojové prostředí Qt Creator. Obsahuje nástroje pro tvorbu GUI, mnoho vlastních knihoven, ke kterým je dostupná srozumitelná nápověda a programy vytvořené v Qt lze kompilovat pro operační systémy Windows na architektuře x86 a pro Linux na architekturách x86 a ARM.

Platforma Linux byla zvolena pro svoji stabilitu a nízkou hardwarovou náročnost. Vyniká také svojí modularitou. Lze začít s lehkou linuxovou distribucí vybavenou pouze příkazovým řádkem do ní přidávat další komponenty. V případě Windows je systém rovnou vybaven všemi funkcemi a ty nepotřebné ze systému nelze odebrat. Systém Windows ve verzích pro stolní počítače navíc trpí postupným zanášením systému pouhým používáním.



Obrázek 8: Nástroj pro navrhování grafického rozhraní (autor, 2019)

3.2 Architektura modelu

3.3 Palubní počítač

Podobně jako v mnoha realizovaných projektech informačních systémů v ostrém provozu, bude i modelový palubní počítat založen na počítači s operačním systémem Linux a procesorem x86. SW počítače se skládá ze dvou částí.

Hlavní část je samotný program obstarávající komunikaci a práci s daty, jehož tvorba je předmětem této práce. Vedlejší částí jsou databázový server (MySQL) a aplikace pro úpravu dat v této databázi (PhpMyAdmin). Tyto dva počítačové programy jsou volně dostupné, dobře zdokumentované a běžně používané, hlavně pro odvětví webdesignu.

3.4 Princip funkce programu

3.4.1 Zpracování dat jízdních řádů

Pro zjednodušení programu samotného (především práce s daty) byla zvolena Klient-server databázová architektura. Klient i server ovšem běží na stejném stroji. Zpracování dat před zobrazením vyřeší již databázový server po přijetí správně složeného databázového dotazu. Z teoretického hlediska se může zdát, že komunikace s databázovým systémem bude neúměrně prodlužovat reakce systému na požadavky řidiče, avšak pokusy na rané verzi programu ukázaly, že čas potřebný pro zpracování DB dotazu je zanedbatelný. Výhodou ukládání dat do databáze je škálovatelnost. Počet položek je omezený pouze množstvím úložného prostoru zařízení a není nutné používat v hlavním programu dynamickou alokaci paměti pro pole. Vzhledem k možnostem exportu/importu z databázového serveru MySQL je možné vstupní data editovat na počítačích s běžnými nástroji pro práci s databázemi. Není nutné řešit datový formát samotný, pouze konkrétní strukturu tabulek v databázi.

V konečném důsledku se hlavní program stará jen o ovládání, zobrazení dat pro řidiče a o komunikaci pomocí VDV301. Jedno z možných řešení vyřizování HTTP požadavků je instalace dedikovaného serveru určeného pro zobrazování webových stránek, jako jsou Apache nebo lighthttpd. Tyto servery jsou bez pokročilé konfigurace uzpůsobené pouze pro odesílání odpovědí na GET požadavky. To znamená, že by byly použitelné pouze pro proceduru GetStatusDevice, kdy by bylo možné měnit vlastním programem palubního počítače obsah XML souboru v počítači. Tento soubor by pak byl poskytován zařízením, která o něj zažádají. Bohužel by již tímto způsobem nebylo možné zpracovat obsah POST dotazů. Proto je síťová komunikace zajištěna přímo knihovnou v prostředí QT, která umožňuje přímo manipulaci s obsahem požadavků GET a POST a odpověďmi na ně.

V programu není implementován systém přidávání odběratelů pomocí metody Subscribe. Odběratele dat je nutné přidat manuálně do zdrojového kódu před kompilací. Poté na předprogramované odběratele odesílá data metodou POST. Data jsou poskytována také pro získání metodou GET na adrese `http://127.0.0.1:47474/CustomerInformationService/GetAllData`

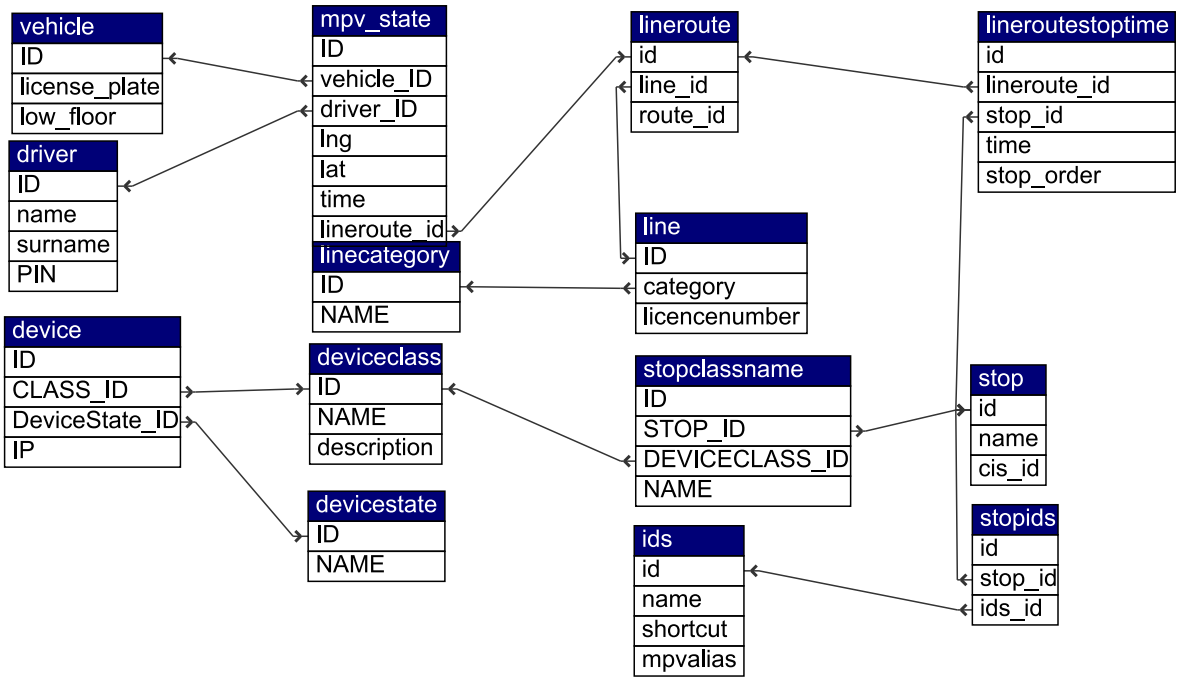
3.4.2 Seznam využitých entit

Databázová struktura je navržena s maximální snahou o pokrytí potřeb palubního počítače pro úložiště jízdních řádů. Databázová struktura nerozlišuje uzly a sloupky, neboť tyto nerozlišuje ani zdroj vstupních dat (export ze systém CIS JŘ).

Line Obsahuje základní údaje o lince: licenční číslo a kategorii linky.

Linecategory Obsahuje kategorie linek:

- MHD - Městská
- PAL -Příměstská



Obrázek 9: architektura databáze(Autor, 2019)

ID	category	licencenumber
1	3	820741
2	2	820881
4	1	100370

Tabulka 1: kompletní seznam linek **line**

- DAL -dálková

ID	NAME
1	PAL
2	MHD
3	DAL

Tabulka 2: číselník typů linek **linecategory**

Lineroute Obsahuje linkospoje. Linkospoj charakterizuje konkrétní orientovaný sled zastávek na lince v dané časové poloze.

Lineroutestoptime Řádky této tabulky tvoří kostru jízdního řádu. Udávají, v jaký čas má být vozidlo konkrétního linkospoje na konkrétní zastávce. Hodnota NULL ve sloupci time znamená, že spoj danou zastávkou projíždí. Taková zastávka není uvedena ve výstupu informačního systému.

id	line_id	route_id
1	1	3
2	1	7
3	1	1
76	4	73

Tabulka 3: Tabulka linkospojů **lineroute**

id	lineroute_id	stop_id	time	stop_order
1	1	39530	06:25:00	1
15	1	39535	NULL	2
29	1	39541	06:29:00	3
43	1	39583	06:31:00	4
57	1	40159	06:34:00	5
71	1	39569	06:36:00	6
2	2	39530	06:35:00	1
16	2	39535	NULL	2
30	2	39541	06:39:00	3
44	2	39583	06:42:00	4
58	2	40159	06:45:00	5
72	2	39569	06:46:00	6
86	2	10250	06:50:00	7
100	2	10248	06:52:00	8
3	3	39530	06:40:00	1

Tabulka 4: vzorek tabulky **lineroutestoptime**

Stop Seznam zastávek. V této práci pojem „zastávka“ představuje skupinu sloupků se stejným názvem jízdním řádu, software ASW JŘ pro stejnou množinu využívá označení „uzel“. Současná databázová struktura nerozlišuje jednotlivé sloupky ani směry zastávek.

V praxi se rozdíl ve filozofii datových struktur projevuje hlavně při sledování vozidel pomocí systémů GNSS. Software ASW JŘ umožňuje přiřadit souřadnice každému sloupku. Backoffice FareOn společnosti Mikroelektronika toto neumožňuje a je nutné přiřadit jednu sadu souřadnic zastávce jako celku – tato sada je určena jako těžiště souřadnic všech sloupků dané zastávky/uzlu. Seznam všech zastávek (kromě čísel cisid) je dostupný v povinně zveřejňovaném balíku JDF souborů na adrese `ftp://ftp.cisjr.cz/seznamy/`.

ids Číselník integrovaných dopravních systémů. Slouží hlavně pro zařazení zastávek do sekcí systému MPVnet. V dotazu na API MPVnetu je nutné parametrem (mpvalias) specifikovat, do kterého systému zastávka spadá. Základní číselník je získán z veřejně dostupných stránek `https://www.cisjr.cz/doc/ids.htm` , zkratky pro parametr z URL adres veřejné části MPVnetu `http://mpvnet.cz`. např. `https://mpvnet.cz/pid/tab` , `https://mpvnet.cz/zlin/tab`

id	name	cis_id
25822	Praha AB„Ústav mateřství	NULL
25823	Praha AB„Želivského	NULL
25824	Praha„Albertov	58936
25825	Praha„Ametystová	58937
25826	Praha„Amforová	47715
25827	Praha„Anděl	58759
25828	Praha„Antala Staška	57405

Tabulka 5: vzorek tabulky **stop**

id	name	shortcut	mpvalias
111	Pražská integrovaná doprava	PID	PID
421	Doprava Ústeckého kraje	DÚK	
512	Integrovaný dopravní systém Libereckého kraje	IDOL	idol
711	Integrovaný dopravní systém Olomouckého kraje	IDSOK	
721	Zlínská integrovaná doprava	ZID	zlin
811	Integrovaný dopravní systém Moravskoslezského kraj...	ODIS	odis

Tabulka 6: vzorek tabulky **ids**

stopids Mezilehlá tabulka modelující vztah m:n mezi zastávkami a číselníkem integrovaných systémů. Jedna zastávka tedy může být přiřazena k více IDS. Ve výsledném programu jsou přestupy stahovány pouze z IDS s nižším ID.

id	stop_id	ids_id
3705	40030	111
3706	40031	111
170	40157	721

Tabulka 7: vzorek mezilehlé tabulky **stopids**

3.4.3 Seznam entit pro budoucí rozvoj

Stopclassname Tabulka umožňující přidělit každé třídě zařízení jiný název zastávky. (Např. Potřeba zkráceného názvu pro tisk na znehodnocovači jízdních dokladů.)

Deviceclass Číselník typů zařízení uvedených v normě VDV301.

Device Tabulka umožňující evidovat aktuálně připojená zařízení k palubnímu počítači. Nabízí se budoucí využití jako seznam odběratelů pro službu SubscribeAllData.

ID	NAME	description
1	OnboardUnit	The central unit of the former IBIS standard
2	Validator	Mechanical or electrical automaton to validate tickets
3	SideDisplay	External display of the side of the vehicle
4	FrontDisplay	External display on front of the vehicle(showing destination)
5	InteriorDisplay	Displays found inside vehicle, from simple text displays to modern TFT displays
6	TicketVendingMachine	A machine for issuing tickets, inside the vehicle, driver- or passenger operated
7	AnnouncementSystem	The electro-acoustical device inside the vehicle
8	MMI	The interface operated by the driver, e-g- a touch display
9	VideoSystem	IBIS-IP controlled video surveillance system
10	APC	Passenger counting on the vehicle
11	MobileInterface	Interface to passenger's mobile devices
12	TestDevice	Placeholder for devices not covered by the standard used during test (development or field)
13	Other	Placeholder for devices not covered by the standard used during test (development or field)

Tabulka 8: kompletní tabulka **deviceclass**

ID	CLASS_ID	DeviceState_ID	IP
5	3	3	192.168.0.21
8	5	1	192.168.0.24
9	1	1	127.0.0.1

Tabulka 9: Tabulka zařízení **device**

Devicestate Číselník provozních stavů, kterých daná zařízení mohou dle VDV301 dosahovat.

Driver Slouží pro evidenci jednotlivých řidičů, pro potřeby dispečerského řízení/zamezení přístupu neoprávněných osob do palubního počítače informačního systému.

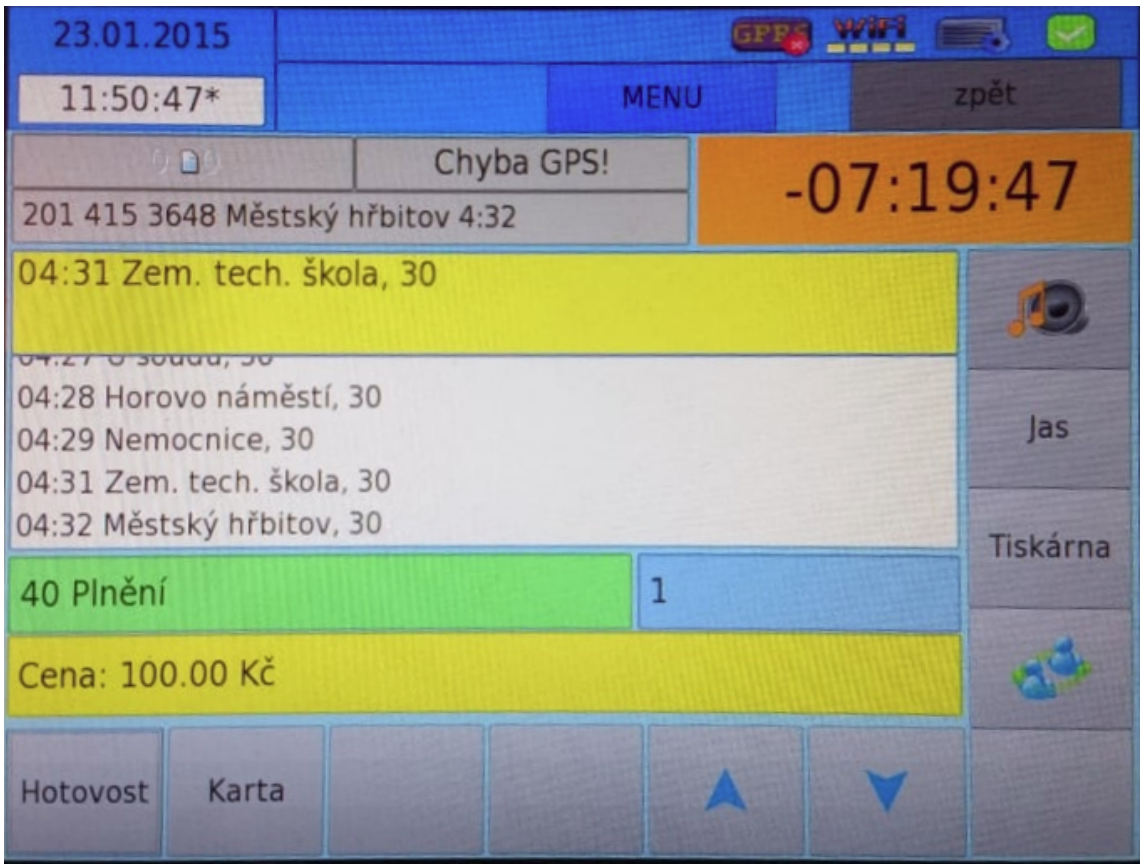
Vehicle Seznam všech vozidel dopravce - potenciální použití pro systémy sledování polohy.

MPV_state Příprava pro systémy AVL (Automatic Vehicle Location). Slouží pro evidenci polohy, linkospoje, řidiče vybraného vozidle v čase. Význam sběru dat je buď zpětná analýza offline dat, nebo průběžné odesílání polohy do systémů dispečerského řízení, jako například MPVnet využívaný v prostředí Pražské Integrované dopravy. Systém záznamu polohy není implementován.

ID	NAME
1	defective
2	notavailable
4	running

Tabulka 10: kompletní tabulka **devicestate**

3.4.4 Popis prvků uživatelského rozhraní



Obrázek 10: obrazovka zařízení EmTest EM126i EMX7 (Autor, 2015)

Provedení grafického rozhraní hlavního programu je maximálně podřízeno funkci. Rozložení ovládacích prvků napodobuje skutečná zařízení s dotykovým displejem určená pro ostrý provoz. Ve srovnání s nimi obsahuje i několik ovládacích prvků navíc. Tyto prvky jsou určeny pro ladění aplikace a diagnostiku propojení se serverem.

SQL Connect inicializuje spojení s databázovým serverem. Adresa serveru je ve zdrojovém kódu nastavena na 127.0.0.1, neboť se nepředpokládá využití externího DB serveru. Stav připojení k SQL serveru je indikován popiskem nad tlačítkem.

Linka, Spoj textová pole, přizpůsobeno pro připojení externí číslicové klávesnice pro manuální zadání čísla linky a spoje. Funkce našeptávání ani výběru není implemento-

vána. Dostupné linky: 820741, 820831,820881, 100370 (PID)

POTVRDIT – po stisku tlačítka je v případě existujícího spoje a linky složen SQL dotaz na server a dojde k vypsání prvních pěti zastávek od začátku linky s pořadovým číslem a časem odjezdu dle pravidelného jízdního řádu.

Přidat a Ubrat Tato tlačítka slouží k posouvání indexu aktuální zastávky bez změny stavu polohy vozidla (locationState).

Šipka vzhůru Krok vpřed při simulaci jízdy. Ovlivňuje stav polohy vozidla (localState) následujícím způsobem:

BeforeStop→AtStop→AfterStop→BetweenStop

Ke zvýšení indexu aktuální zastávky dochází při přechodu mezi AtStop a AfterStop.

Šipka dolů Krok zpět při simulaci jízdy. Ovlivňuje stav polohy vozidla (localState) následujícím způsobem:

BetweenStop→AfterStop→AtStop→BeforeStop

Ke snížení indexu aktuální zastávky dochází při přechodu mezi AfterStop a AtStop.

QUIT Okamžitě ukončí program.

Dveřní volič Umožňuje nasimulovat všechny dveřní stavy dovolené normou.

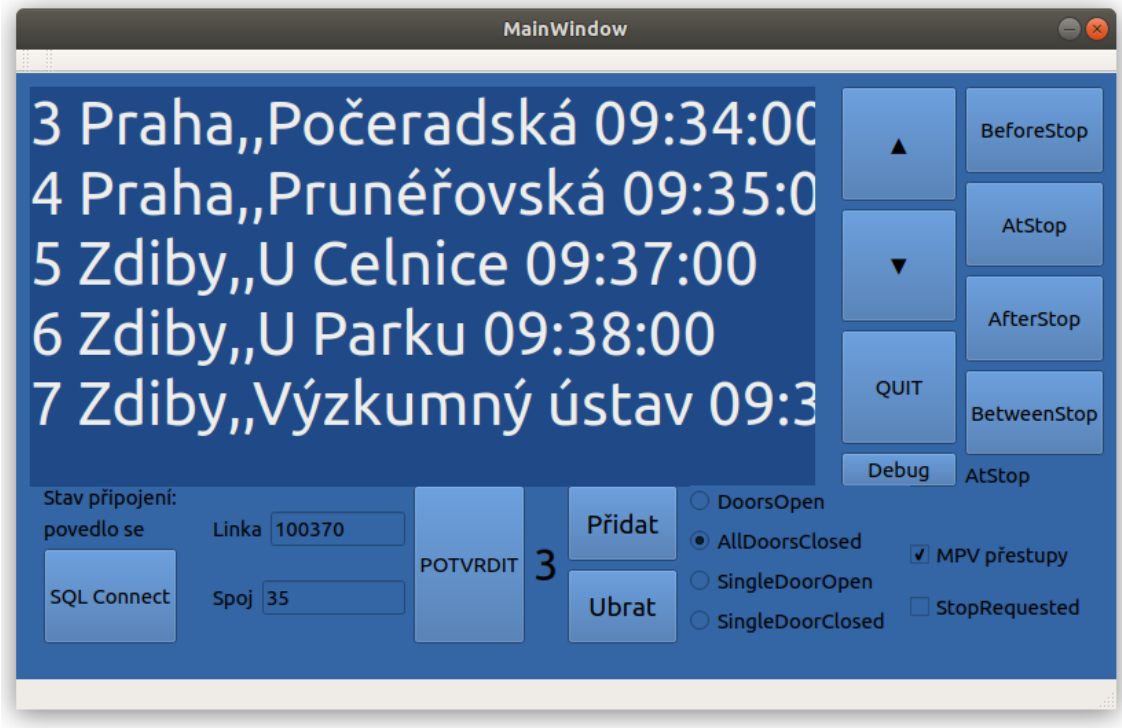
Zatržítko MPV přestupy Pokud je zatržítko ve stavu vypnuto, vůbec nedochází ke stahování aktuálních dat ze serveru MPVnet a tedy ani k zahrnutí předchozích stažených dat do XML pro zobrazení na periferiích.

Zatržítko StopRequest Slouží pro testování reakce periferií na stisk tlačítka "na znamení".

Tlačítko Debug Funkce tohoto tlačítka je proměnná, dle aktuální potřeby dalšího vývoje.

Voliče Polohy Tlačítka BeforeStop, AtStop, AfterStop a BetweenStop slouží pro přímou volbu stavu polohy (locationState).Nemění index aktuální zastávky. Aktuálně zvolený stav polohy je indikován pomocí popisu pod těmito tlačítky.

Zadávání linky a spoje je řešeno textovými poli a není narozdíl od zbytku rozhraní přívětivé pro ovládání prstem přes dotykové displeje.



Obrázek 11: Uživatelské rozhraní OBU

3.5 MPVnet přestupy

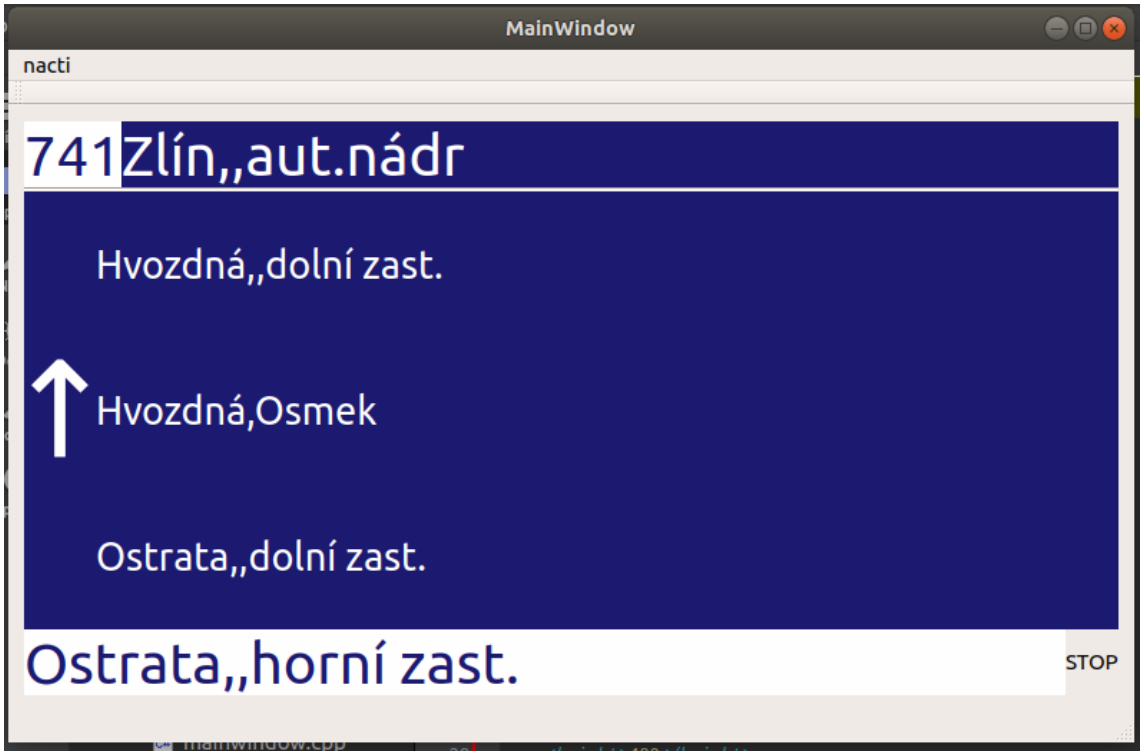
Jedním z požadavků regionálního organizátora PID je poskytování informací o přestupu na ostatní linky MHD odjíždějící z následující zastávky. Tyto informace by bylo možné získávat vyhledáváním v databázi jízdních řádů všech linek odjíždějících z daného stanoviště. V případě zobrazování na LCD v autobusu by však palubní počítač musel obsahovat i kompletní jízdní řády linek metra. Z tohoto a dalších důvodů jsou proto informace o odjezdech generovány on-line z dispečerského systému MPVnet, který navíc poskytuje informace o reálném zpoždění jednotlivých spojů. Informace o reálné poloze a tedy ani o zpoždění v současnosti nejsou poskytovány u vozidel Dopravního podniku hlavního města Prahy.

Způsob přístupu k datům o přestupech ze systému MPVnet bude v praxi řešen velice podobně jako v této práci, tedy přes neveřejné API poskytující data v XML formátu, jako v následujícím příkladu. Pro zobrazení odjezdů z konkrétní zastávky je nutné znát pořadové číslo zastávky odpovídající systému CIS JŘ (Celostátní informační systém o jízdních řádech). Číselník všech zastávek v systému opět není uživatelsky dostupný. Prohlížet stejná data o odjezdech v uživatelsky přívětivém formátu lze ve webové aplikaci `http://mpvnet.cz/pid/tabule`.

Strojově čitelný vzorový výstup z neveřejného API:

```
1 <TBL cas="2019-08-10T23:12:41" ver="1.0.7145.21217" text="Ověřovací␣
    provoz.␣Bez␣záruky.">
2 <t id="62887" stan="A,B,M1,M2" zast="Národní␣třída">
3 <o stan="A" lin="9" alias="9" spoj="77" smer="Praha,Spojovací" odj="
    2019-08-10T23:16:00+02:00" sled="false" zpoz="0" np="true" nad="
    false" t="Tram" dd="2" smer_c="27891"/>
```

```
<o stan="B" lin="18" alias="18" spoj="15" smer="Praha,Nádraží␣Podbaba"
    odj="2019-08-10T23:16:00+02:00" sled="false" zpoz="0" np="true"
   nad="false" t="Tram" dd="2" smer_c="63414"/>
<o stan="A" lin="22" alias="22" spoj="273" smer="Praha,Nádraží␣Straš
   nice" odj="2019-08-10T23:16:00+02:00" sled="false" zpoz="0" np="
   true" nad="false" t="Tram" dd="2" smer_c="57696"/>
<o stan="M1" lin="B" alias="B" spoj="32" smer="Praha,Zličín" odj="
   2019-08-10T23:16:00+02:00" sled="false" zpoz="0" np="false" nad="
   false" t="Metro" dd="1" smer_c="28037"/>
<o stan="B" lin="22" alias="22" spoj="161" smer="Praha,Bílá␣Hora" odj=
   "2019-08-10T23:17:00+02:00" sled="false" zpoz="0" np="true" nad="
   false" t="Tram" dd="2" smer_c="27908"/>
</t>
</TBL>
```



Obrázek 12: základní obrazovka modelového LCD panelu

Obrázek 13: obrazovka skutečného LCD společnosti BUSTEC

Obrázek 14: obrazovka skutečného LCD s upravenou grafikou (autor, 2019)

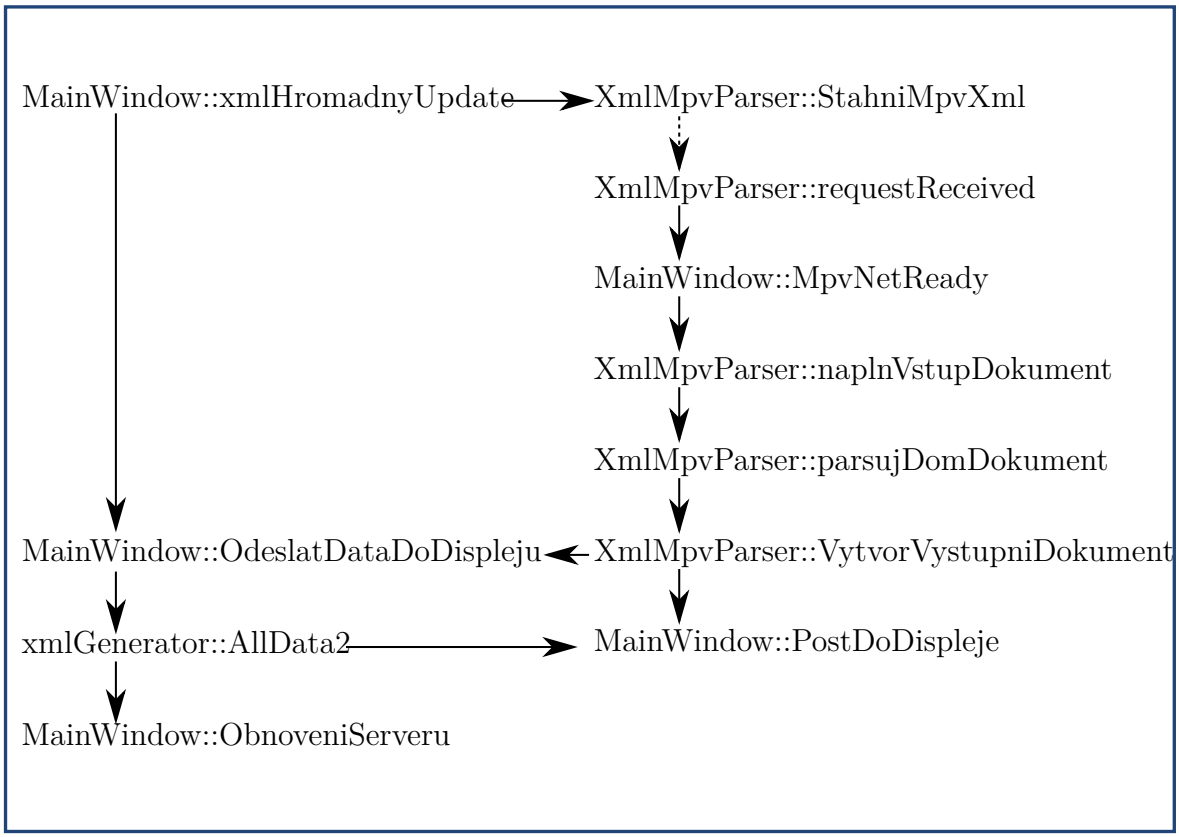
Obrázek 15: obrazovka skutečného LCD s upravenou grafikou (autor, 2019)

4 Ukázky zdrojového kódu s popisem

4.1 VDV301_OBU

4.1.1 cestaudaje.h

Tato třída sdružuje všechny globální proměnné využívané pro zobrazení, zpracování



Obrázek 16: Posloupnost funkcí při aktualizaci dat pro OIS (autor, 2019)

Kromě rozsáhlejších dat uložených v databázi disponuje program i svými globálními proměnnými.

Tyto globální proměnné udržují stavové informace celého systému potřebné pro zobrazování a zpracování jízdního řádu.

Jde hlavně o polohu vozidla vůči zastávce, počet zastávek, index aktuální zastávky.

```
#include <QMainWindow>
#include <QObject>
#include <QWidget>

class CestaUdaje : public QMainWindow
{
    Q_OBJECT
public:
    explicit CestaUdaje(QWidget *parent = nullptr);
    QString doorState="AllDoorsClosed";
    QString locationState="AtStop";
    bool VehicleStopRequested=false;
```

```
    bool prestupy=false;
    int cislo;
    int pocetZastavek;
    int aktlinka;
    int aktspoj;
    int cisloVozu;
    int otevreneDvere;
    int vystupniStrana;
signals:
public slots:
};
```

4.1.2 mainwindow.h

Hlavní třída celého programu. Staré se o interakci s uživatelským rozhraním a propojuje ostatní vstupy.

```
aktualizace dat dostupných na \url{http://127.0.0.1:47474/
    CustomerInformationService/GetAllData}
void MainWindow::ObnoveniServeru(QByteArray dataDoServeru)
{
    QByteArray hlavicka="";
    QByteArray argumentXMLserveru = "";
    hlavicka+=("HTTP/1.1 200 OK\r\n");       // \r needs to be before
        \n
    hlavicka+=("Content-Type: application/xml\r\n");
    hlavicka+=("Connection: close\r\n");
    hlavicka+=("Pragma: no-cache\r\n");
    hlavicka+=("\r\n");

    argumentXMLserveru.append(hlavicka);
    argumentXMLserveru.append(dataDoServeru);
    HHserver.zapisDoPromenne(argumentXMLserveru);
}
```

odeslání dat do konkrétního zařízení metodou POST

```


void MainWindow::PostDoDispleje(QUrl adresaDispleje, QByteArray
    dataDoPostu)
{
    QByteArray postDataSize = QByteArray::number(dataDoPostu.size());
    QNetworkRequest pozadavekPOST(adresaDispleje);
    pozadavekPOST.setRawHeader("Connection", "Keep-Alive");
    pozadavekPOST.setRawHeader("Content-Length", postDataSize );
    pozadavekPOST.setRawHeader("Content-Type", "text/xml");
    // pozadavekPOST.setRawHeader("Expect", "100-continue");
    pozadavekPOST.setRawHeader("Accept-Encoding", "gzip, deflate");
    QNetworkAccessManager *manager2 = new QNetworkAccessManager();
    manager2->post(pozadavekPOST,dataDoPostu);
}
```

4.1.3 prestupmpv.h

Datová struktura pro dočasné uložení údajů o jednom přestupu. V praxi je v programu využita jako pole přestupů o pevné velikosti, jelikož parametr v požadaveku na MPVnet obsahuje omezení na 15 přestupů.

```
class prestupMPV
{
public:
    prestupMPV();
    QString stan="";
    QString lin="";
    QString alias="";
    int spoj=0;
    QString smer="";
    QString odj="";
    int zpoz=0;
    bool np=false;
    bool nad=false;
    QString t="";
    int dd=0;
    int smer_c=0;
    bool sled=false;
};
```

4.1.4 sqlprace.h

Třída sqlprace zajišťuje kompletní práci s databází od připojení až po vytvoření dotazu a uložení dat do dočasných datových struktur. [2]

```

void SQLprace::StahniSeznam(int &pocetVysledku, int cisloLinky, int
    cisloSpoje, SeznamZastavek *docasnySeznamZastavek )
{
    qDebug()<< "SQLprace::StahniSeznam";
    qInfo()<<"DebugPointA";
    QString queryString("SELECT a.stop_order, b.name, a.time, b.cis_id
        ,f.mpvalias FROM lineroutestoptime a ");
    queryString+=(" LEFT JOIN  stop b ON b.id = a.stop_id ");
    queryString+=(" LEFT JOIN lineroute c ON a.lineroute_id=c.id ");
    queryString+=(" LEFT JOIN line d ON c.line_ID = d.ID ");
    queryString+=(" LEFT JOIN stopids e ON b.id = e.stop_id ");
    queryString+=(" LEFT JOIN ids f ON e.ids_id=f.id ");
    queryString+=(" WHERE d.licencenumber=");
    queryString+=( QString::number(cisloLinky));
    queryString+=( " AND c.route_id =");
    queryString+=( QString::number(cisloSpoje));
    /* queryString+=( " AND a.stop_order>=");
      queryString+=( QString::number(cisloporadi));  */
    queryString+=("  ORDER BY a.stop_order");
    //queryString+="  ASC LIMIT 5");
    QSqlQuery query(queryString,this->mojeDatabaze);
    int counter=0;
    qDebug()<<"DebugPointB";
    int citacMaximum=0;
    while (query.next())
```

```
26      {
27          if (query.value(1).toString()!="")
28          {
29
30              int cisloZast = query.value(0).toInt();
31              if (cisloZast>=citacMaximum)
32              {
33                  citacMaximum=cisloZast;
34              }
35              docasnySeznamZastavek[cisloZast].StopIndex=cisloZast;
36              QString cisloZastString = QString::number(cisloZast);
37              qDebug()<<query.value(0).toString();
38              QString jmenoZastavky = query.value(1).toString();
39              docasnySeznamZastavek[cisloZast].StopName= jmenoZastavky;
40              QString casPrijezdu = query.value(2).toString();
41              docasnySeznamZastavek[cisloZast].cisloCis=query.value(3).
                    toInt();
42              docasnySeznamZastavek[cisloZast].ids =query.value(4).
                    toString();
43              qInfo()<<"DebugPointC";
44              counter++;
45              qDebug()<<"citac:␣"<<citacMaximum ;
46          }
47      }
48      counter=citacMaximum;
49      qInfo()<<"DebugPointD"<<"velikost␣counteru␣je␣"<<QString::number(
            counter);
50      QString cil=docasnySeznamZastavek[counter].StopName;
51
52      for (int i=1; i<=counter;i++)
53      {
54          docasnySeznamZastavek[i].DestinationName=cil;
55      }
56      qInfo()<<"DebugPointF";
57      VypisPole(docasnySeznamZastavek,counter);
58      pocetVysledku=counter;
59      qInfo()<<"pocetzastavek␣je"<<QString::number(pocetVysledku);
60  }
```

4.1.5 xmlgenerator.h

Třída xmlgenerator vytváří obsah pro službu *CustomerInformationService/GetAllData.xml*, tzn. samotné XML k odeslání.

```
1  QByteArray xmlGenerator::AllData2(int poradi, int pocetZastavek,
       SeznamZastavek *docasnySeznamZastavek, int docasLinka, QString
       doorState, QString locationState, QDomDocument Connections)
2  {
3      qDebug()<<"␣xmlGenerator::AllData2␣";
4
5      QByteArray vysledekMpvnetu ="";
6      QString testVysledek="<TBL␣cas=\"2019-08-10T23:12:41\"␣ver
           =\"1.0.7145.21217\"␣text=\"Ověřovací␣provoz.␣Bez␣záruky.\"><t␣
           id=\"62887\"␣stan=\"A,B,M1,M2\"␣zast=\"Národní␣třída\"><o␣stan
           =\"A\"␣lin=\"9\"␣alias=\"9\"␣spoj=\"77\"␣smer=\"Praha,Spojovac
           í\"␣odj=\"2019-08-10T23:16:00+02:00\"␣sled=\"false\"␣zpoz
           =\"0\"␣np=\"true\"␣nad=\"false\"␣t=\"Tram\"␣dd=\"2\"␣smer_c
```

```
        =\"27891\"/><o␣stan=\"B\"␣lin=\"18\"␣alias=\"18\"␣spoj=\"15\"␣
        smer=\"Praha,Nádraží␣Podbaba\"␣odj=\"2019-08-10T23
        :16:00+02:00\"␣sled=\"false\"␣zpoz=\"0\"␣np=\"true\"␣nad=\"
        false\"␣t=\"Tram\"␣dd=\"2\"␣smer_c=\"63414\"/><o␣stan=\"A\"␣
        lin=\"22\"␣alias=\"22\"␣spoj=\"273\"␣smer=\"Praha,Nádraží␣Stra
        šnice\"␣odj=\"2019-08-10T23:16:00+02:00\"␣sled=\"false\"␣zpoz
        =\"0\"␣np=\"true\"␣nad=\"false\"␣t=\"Tram\"␣dd=\"2\"␣smer_c
        =\"57696\"/><o␣stan=\"M1\"␣lin=\"B\"␣alias=\"B\"␣spoj=\"32\"␣
        smer=\"Praha,Zličín\"␣odj=\"2019-08-10T23:16:00+02:00\"␣sled
        =\"false\"␣zpoz=\"0\"␣np=\"false\"␣nad=\"false\"␣t=\"Metro\"␣
        dd=\"1\"␣smer_c=\"28037\"/><o␣stan=\"B\"␣lin=\"22\"␣alias
        =\"22\"␣spoj=\"161\"␣smer=\"Praha,Bílá␣Hora\"␣odj=\"2019-08-10
        T23:17:00+02:00\"␣sled=\"false\"␣zpoz=\"0\"␣np=\"true\"␣nad=\"
        false\"␣t=\"Tram\"␣dd=\"2\"␣smer_c=\"27908\"/></t></TBL>";
7
8
9       qDebug()<<"␣␣";
10      QByteArray language="cz";
11      QByteArray deflanguage="cz";
12      QByteArray lineNumber=QByteArray::number(docasLinka);
13      QByteArray lineName=lineNumber.right(3);
14      QByteArray vehicleref="33";
15      QByteArray currentStopIndex= QByteArray::number(poradi);
16      QByteArray routeDeviation="onroute";
17      QByteArray vehicleStopRequested="0";
18      QByteArray exitSide="right";
19      QByteArray tripRef="3";
20      QByteArray displayContentRef="1234";
21      QByteArray destinationRef="22";
22      QByteArray destinationName="";
23      destinationName+=docasnySeznamZastavek[poradi].DestinationName;
24      qDebug()<<"nazev␣cile2"<<destinationName;
25      QByteArray dotaz="";
26      QByteArray hlavicka="";
27      QByteArray telo="";
28      QDomDocument xmlko;
29      QDomProcessingInstruction dHlavicka=xmlko.
           createProcessingInstruction("xml","version=\"1.0\"␣encoding=\"
           utf-8\"␣");
30      xmlko.appendChild(dHlavicka);
31      QDomElement dCustomerInformationService=xmlko.createElement("
           CustomerInformationService.GetAllDataResponse");
32      QDomElement dAllData=xmlko.createElement("AllData");
33      xmlko.appendChild(dCustomerInformationService);
34      dCustomerInformationService.appendChild(dAllData);
35      QDomElement dTimeStamp=xmlko.createElement("TimeStamp");
36      dTimeStamp.setNodeValue(this->createTimestamp());
37      dAllData.appendChild(dTimeStamp);
38      QDomElement dVehicleRef=xmlko.createElement("VehicleRef");
39      dVehicleRef.appendChild(xmlko.createElement("Value"));
40      dVehicleRef.firstChildElement("Value").appendChild(xmlko.
           createTextNode(vehicleref));
41      dAllData.appendChild(dVehicleRef);
42      QDomElement dDefaultLanguage=xmlko.createElement("DefaultLanguage"
           );
43      dDefaultLanguage.appendChild(xmlko.createElement("Value"));
44      dDefaultLanguage.firstChildElement("Value").appendChild(xmlko.
           createTextNode(deflanguage));
```

```
45      dAllData.appendChild(dDefaultLanguage);
46
47      QDomElement dTripInformation=xmlko.createElement("TripInformation"
            );
48      dAllData.appendChild(dTripInformation);
49
50      QDomElement dLocationState=xmlko.createElement("LocationState");
51      dLocationState.appendChild(xmlko.createTextNode( locationState));
52      dTripInformation.appendChild(dLocationState);
53
54      QDomElement dTripRef=xmlko.createElement("TripRef");
55      dTripRef.appendChild(xmlko.createElement("Value"));
56      dTripRef.firstChildElement("Value").appendChild(xmlko.
            createTextNode(tripRef));
57      dTripInformation.appendChild(dTripRef);
58
59      QDomElement dStopSequence=xmlko.createElement("StopSequence");
60      dTripInformation.appendChild(dStopSequence);
61      for (int i=1 ; i<=pocetZastavek;i++)
62      {
63          QByteArray cCurrentStopIndex=QByteArray::number(
                docasnySeznamZastavek[i].StopIndex);
64          QByteArray cStopName="";
65          cStopName+= docasnySeznamZastavek[i].StopName;
66          QByteArray xDestination="";
67          QByteArray xDestinationRef="";
68          QByteArray xDestinationName="";
69          xDestinationName+= docasnySeznamZastavek[i].DestinationName;
70          //STOP
71          QDomElement dStopPoint=xmlko.createElement("StopPoint");
72          dStopSequence.appendChild(dStopPoint);
73          QDomElement dStopIndex=xmlko.createElement("StopIndex");
74          dStopIndex.appendChild(xmlko.createElement("Value"));
75          dStopIndex.firstChildElement("Value").appendChild(xmlko.
                createTextNode(cCurrentStopIndex ));
76          dStopPoint.appendChild(dStopIndex);
77
78          QDomElement dStopRef=xmlko.createElement("StopRef");
79          dStopRef.appendChild(xmlko.createElement("Value"));
80          dStopRef.firstChildElement("Value").appendChild(xmlko.
                createTextNode("Ref:"+cCurrentStopIndex));
81          dStopPoint.appendChild(dStopRef);
82
83          QDomElement dStopName=xmlko.createElement("StopName");
84          dStopName.appendChild(xmlko.createElement("Value"));
85          dStopName.firstChildElement("Value").appendChild(xmlko.
                createTextNode(cStopName));
86          dStopName.appendChild(xmlko.createElement("Language"));
87          dStopName.firstChildElement("Language").appendChild(xmlko.
                createTextNode(language));
88          dStopPoint.appendChild(dStopName);
89
90          QDomElement dDisplayContent=xmlko.createElement("
                DisplayContent");
91          dStopPoint.appendChild(dDisplayContent);
92
93          QDomElement dDisplayContentRef=xmlko.createElement("
                DisplayContentRef");
```

```
94          dDisplayContentRef.appendChild(xmlko.createElement("Value"));
95          dDisplayContentRef.firstChildElement("Value").appendChild(
                xmlko.createTextNode("1234"));
96          dDisplayContent.appendChild(dDisplayContentRef);
97
98          QDomElement dLineInformation=xmlko.createElement("
                LineInformation");
99          dDisplayContent.appendChild(dLineInformation);
100         QDomElement dLineRef=xmlko.createElement("LineRef");
101         dLineRef.appendChild(xmlko.createElement("Value"));
102         dLineRef.firstChildElement("Value").appendChild(xmlko.
                createTextNode(lineNumber));
103         dLineInformation.appendChild(dLineRef);
104         QDomElement dLineName=xmlko.createElement("LineName");
105         dLineName.appendChild(xmlko.createElement("Value"));
106         dLineName.firstChildElement("Value").appendChild(xmlko.
                createTextNode(lineName));
107         dLineInformation.appendChild(dLineName);
108         dLineName.appendChild(xmlko.createElement("Language"));
109         dLineName.firstChildElement("Language").appendChild(xmlko.
                createTextNode(language));
110         QDomElement dLineNumber=xmlko.createElement("LineNumber");
111         dLineNumber.appendChild(xmlko.createElement("Value"));
112         dLineNumber.firstChildElement("Value").appendChild(xmlko.
                createTextNode(lineNumber));
113         dLineInformation.appendChild(dLineNumber);
114         QDomElement dDestination=xmlko.createElement("Destination");
115         dDisplayContent.appendChild(dDestination);
116         QDomElement dDestinationRef=xmlko.createElement("
                DestinationRef");
117         dDestinationRef.appendChild(xmlko.createElement("Value"));
118         dDestinationRef.firstChildElement("Value").appendChild(xmlko.
                createTextNode(destinationRef));
119         dDestination.appendChild(dDestinationRef);
120
121         QDomElement dDestinationName=xmlko.createElement("
                DestinationName");
122         dDestinationName.appendChild(xmlko.createElement("Value"));
123         dDestinationName.firstChildElement("Value").appendChild(xmlko.
                createTextNode(destinationName));
124         dDestinationName.appendChild(xmlko.createElement("Language"));
125         dDestinationName.firstChildElement("Language").appendChild(
                xmlko.createTextNode(language));
126         dDestination.appendChild(dDestinationName);
127
128         if (cCurrentStopIndex==currentStopIndex)
129         {
130             qDebug()<<"␣prestupy␣"<<Connections.toString();
131
132             QDomNodeList seznamPrestupu = Connections.
                    elementsByTagName("Connection");
133             for (int j=0;j<seznamPrestupu.count();j++)
134             {
135                 dStopPoint.appendChild(seznamPrestupu.at(i).toElement
                        () );
136             }
137         }
138     }
```

```
139
140     QDomElement dCurrentStopIndex=xmlko.createElement("
            CurrentStopIndex");
141     dCurrentStopIndex.appendChild(xmlko.createElement("Value")).
            appendChild(xmlko.createTextNode(currentStopIndex));
142     dAllData.appendChild(dCurrentStopIndex);
143     QDomElement dRouteDeviation = xmlko.createElement("RouteDeviation"
            );
144     QDomElement dDoorState = xmlko.createElement("DoorState");
145     dDoorState.appendChild(xmlko.createTextNode(doorState));
146     dAllData.appendChild(dDoorState);
147     QDomElement dVehicleStopRequested=xmlko.createElement("
            VehicleStopRequested");
148     dVehicleStopRequested.appendChild(xmlko.createElement("Value")).
            appendChild(xmlko.createTextNode(vehicleStopRequested));
149     dAllData.appendChild(dVehicleStopRequested);
150     QDomElement dExitSide = xmlko.createElement("ExitSide");
151     dExitSide.appendChild(xmlko.createTextNode(exitSide));
152     dAllData.appendChild(dExitSide);
153     telo="";
154     telo+=xmlko.toString();
155     return telo;
156 }
```

4.1.6 xmlmpvparser.h

Třída xmlmpvparser.h získává informace o přestupech pomocí HTTP GET z MPVnet API ve formě XML. Toto xml má naprosto jinou vnitřní strukturu než přestupy potřebné pro službu GetAllData. Proto pomocí funkcí v této třídě dojde nejdřív k vyparsování dat z XML pomocí DOM příkazů a následně je z vyparsovaných dat vytvořena XML formátovaná část výsledného XML pro službu GetAllData.

Informace o přestupech jsou získávány vždy jen pro aktuální zastávku z důvodu snížení zátěže na MPVnet API a to v situacích, kdy dochází ke změně indexu aktuální zastávky, případě ke změně polohy vůči zastávce. Celou funkčnost této třídy lze deaktivovat pomocí zatržítka v uživatelském rozhraní.

```
1  int XmlMpvParser::parsujDomDokument()
2  {
3      qDebug()<<"parsujDomDokument";
4      QDomElement root = vstupniDomDokument.firstChildElement();
5      QDomNodeList nodes = root.elementsByTagName("t").at(0).toElement()
           .elementsByTagName("o");
6      int docasnyPocetPrestupu= nodes.count();
7      for (int i=0; i<nodes.count();i++)
8      {
9          mujPrestup[i].stan=nodes.at(i).attributes().namedItem("stan").
               nodeValue();
10         mujPrestup[i].lin=nodes.at(i).attributes().namedItem("lin").
               nodeValue();
11         mujPrestup[i].alias=nodes.at(i).attributes().namedItem("alias"
               ).nodeValue();
12         mujPrestup[i].spoj=nodes.at(i).attributes().namedItem("spoj").
               nodeValue().toInt();
```

```
13          mujPrestup[i].smer=nodes.at(i).attributes().namedItem("smer").
                nodeValue();
14          mujPrestup[i].odj=nodes.at(i).attributes().namedItem("odj").
                nodeValue();
15          mujPrestup[i].zpoz=nodes.at(i).attributes().namedItem("zpoz").
                nodeValue().toInt();
16          mujPrestup[i].sled=nodes.at(i).attributes().namedItem("sled").
                nodeValue().toInt();
17          mujPrestup[i].np=nodes.at(i).attributes().namedItem("np").
                nodeValue().toInt();
18          mujPrestup[i].nad=nodes.at(i).attributes().namedItem("nad").
                nodeValue().toInt();
19          mujPrestup[i].t=nodes.at(i).attributes().namedItem("t").
                nodeValue();
20          mujPrestup[i].dd=nodes.at(i).attributes().namedItem("dd").
                nodeValue().toInt();
21          mujPrestup[i].smer_c=nodes.at(i).attributes().namedItem("
                smer_c").nodeValue().toInt();
22          qDebug()<<mujPrestup[i].smer;
23      }
24      return docasnyPocetPrestupu;
25  }
```

Vytváření XML s přestupy spojováním řetězců Tato metoda se jeví jako ideální pro začínající programátory. Nevyžaduje příliš mnoho přemýšlení a učení se nových funkcí. Na druhou stranu vyžaduje zvýšenou pozornost při převádění tagů do textové podoby, tzn. je nutné kontrolovat shodu otevírajícího i zavírajícího tagu a správné vyplnění znaků <,/,>. Tato metoda je neudržitelná pro tvorbu rozsáhlých dokumentů.

```
1  vysledek+=mujPrestup[i].smer;
2  vysledek+="</Value><Language>cz</Language></DestinationName></
       Destination></DisplayContent><Platform><Value>";
3  vysledek+=mujPrestup[i].stan;
4  vysledek+="</Value></Platform><ConnectionMode>";
5  vysledek+=dopravniProstredek;
6  vysledek+="</ConnectionMode><ExpectedDepatureTime><Value>";
7  vysledek+=mujPrestup[i].odj;
8  vysledek+="</Value></ExpectedDepatureTime></Connection>";
```

Vytváření XML s přestupy pomocí metody DOM Tato metoda je speciálně určená pro tvorbu dokumentů označených pomocí značkovacích jazyků podobných strukturou XML jazyka, např. HTML.

Tato metoda nepracuje s textem, ale s objekty. Na text je obsah převeden až v posledním kroku. Vytvoření tagu probíhá automaticky, je tedy nutné kontrolovat jeho název jen na jednom místě a uzavírací tag je vytvořen automaticky.

Tato metoda je vhodná pro tvorbu rozsáhlých dokumentů a umožňuje rychlé změny ve struktuře dokumentu narozdíl od metody spojování řetězců. Vyžaduje však hlubší uvažování o datových typech, které DOM funkce vyžadují jako vstup, případně jakého datového typu je návratová hodnota funkce.

```
1  QDomElement dDestinationName=xmlko.createElement("DestinationName");
```

```
2  dDestinationName.appendChild(xmlko.createElement("Value"));
3  dDestinationName.firstChildElement("Value").appendChild(xmlko.
       createTextNode(mujPrestup[i].smer));
4  dDestination.appendChild(dDestinationName);
5  dDestinationName.appendChild(xmlko.createElement("Language"));
6  dDestinationName.firstChildElement("Language").appendChild(xmlko.
       createTextNode(language));
7  dConnection.appendChild(xmlko.createElement("Platform")).appendChild(
       xmlko.createElement("Value")).appendChild(xmlko.createTextNode(
       mujPrestup[i].stan));
8  dConnectionMode.appendChild(xmlko.createElement("PtMainMode")).
       appendChild(xmlko.createTextNode(mainMode));
9  dConnectionMode.appendChild(xmlko.createElement(mainMode)).appendChild
       (xmlko.createTextNode(subMode));
10 dConnection.appendChild(dConnectionMode);
11 QDomElement dExpectedDepartureTime=xmlko.createElement("
       ExpectedDepatureTime"); //verze 1.0
12  // QDomElement dExpectedDepartureTime=xmlko.createElement("
        ExpectedDepartureTime"); verze 2.0
```

4.2 VDV301_Display

Současná verze je taktéž provozována na počítači s architekturou x86, však bylo by možné ji implementovat i na jednodeskové počítače s architekturou ARM, jako například Raspberry Pi.

Na tomto zařízení běží zjednodušená varianta hlavního programu z palubního počítače. Využívá stejné komponenty pro komunikaci pomocí VDV301, avšak nemusí obsahovat žádné ovládací prvky ani databázový server.

Program přijímá data metodou POST z z programu VDV301_OBU. Automatické přihlašování k odběru není implementováno. Zobrazení přestupních vazeb není implementováno. Ověření funkčnosti generování přestupních vazeb proběhlo na reálném TFT_LCD Bustec.

mainwindow.h V případě tohoto programu se třída mainwindow stará o vykreslování přijatých dat a opět propojuje všechny ostatní třídy.

xmlparser.h Slouží pro načtení a zpracování XML z POST požadavku přijatého z HTTP serveru.

```
1
2
3  void XmlParser::VytvorSeznamZastavek(SeznamZastavek *docasnySeznamZst,
       int *docasnyIndexZastavky, int *docasnyPocetZastavek)
4  {
5      QDomElement root = dokument.firstChildElement();
6      QDomNodeList nodes = root.elementsByTagName("StopPoint");
7      qInfo()<<"testAVVCF";
8      *docasnyPocetZastavek= nodes.count();
9      *docasnyIndexZastavky=root.elementsByTagName("CurrentStopIndex").
           at(0).firstChildElement().text().toInt();
10     for (int i=0; i<nodes.count();i++)
```

```
11      {
12          int poradiZastavky=root.elementsByTagName("StopPoint").at(i).
               toElement().elementsByTagName("StopIndex").at(0).
               firstChildElement().text().toInt();
13          docasnySeznamZst[poradiZastavky].StopName=root.
               elementsByTagName("StopPoint").at(i).toElement().
               elementsByTagName("StopName").at(0).firstChildElement().
               text();
14          docasnySeznamZst[poradiZastavky].LineName=root.
               elementsByTagName("DisplayContent").at(0).toElement().
               elementsByTagName("LineInformation").at(0).toElement().
               elementsByTagName("LineName").at(0).firstChildElement().
               text();
15          docasnySeznamZst->StopIndex=poradiZastavky;
16          docasnySeznamZst[poradiZastavky].DestinationName=root.
               elementsByTagName("DisplayContent").at(0).toElement().
               elementsByTagName("Destination").at(0).toElement().
               elementsByTagName("DestinationName").at(0).
               firstChildElement().text();
17          qInfo()<< "22" << docasnySeznamZst[i].StopName;
18      }
19  }
```

seznamzastavek.h Plní zde funkci dočasného úložiště dat zpracovaných z XML. Pro potřeby programu je využito pole prvků datového typu SeznamZastavek.

```
1  #ifndef SEZNAMZASTAVEK_H
2  #define SEZNAMZASTAVEK_H
3  #include <QMainWindow>
4  #include <QObject>
5
6  class SeznamZastavek
7  {
8  public:
9      int StopIndex=0;
10     int cisloCis=0;
11     QString ids="";
12     QString StopName="";
13     QString LineName="";
14     QString DestinationName="";
15     SeznamZastavek();
16 };
17
18 #endif // SEZNAMZASTAVEK_H
```

4.2.1 Konfigurace sítě

IP adresy jsou voleny staticky a manuálně přiděleny. Služba mDNS/DNS-SD není implementována.

4.2.2 Tvorba testovacích dat

V rámci zjednodušení práce je vytváření demonstračních dat pouze částečně automatizované a není příliš uživatelsky přívětivé. Jako zdroj využívá data z Centrálního

Název	IP adresa	port	služba
palubní PC	192.168.1.1	47474	poskytování dat pomocí HTTP GET
LCD vlastní	127.0.0.1	47475	příjem dat přes HTTP POST
LCD Bustec	192.168.1.128	60011	příjem dat přes HTTP POST

Tabulka 11: Konfigurace IP adres sestavy

Informačního Systému. Ta jsou veřejně dostupná ve formátu JDF. Dříve byla dostupná i ve formátu XLS, jehož zpracování je pro účely této práce jednodušší. Bohužel, data lze nyní ze stránek portal.idos.cz stáhnout pouze ve formátu PDF a pro tvorbu XLS souborů je tedy využívána cloudová služba *Strojové čtení JDF* autora Radka Papeže dostupná na adrese `https://portal.radekpapez.cz/`. Tento XLS soubor je zpracován pomocí vložení do šablony, která z dat vytvoří SQL příkazy pro vložení do databáze. Všechny tyto kroky je nutné provádět manuálně. Přes počáteční obtíže umožňuje tento postup import kterékoliv linky ze systému CIS JŘ, případně vlastní linky vytvořené v šabloně linky stávající.

1	**Linka číslo 820741**		Zlín-Hvozdná-O	line id	1	
2	**Platí od 11. 12. 2016 do 9. 12. 2017**		INSERT INTO linerout	INSERT INTO lineroute (line_id	INSERT INTO	INSERT IN
3	Tč			Spoj 3	Spoj 7	Spoj 1
4				6 34	X	X 14
5	1	Zlín,,aut.nádr.	odjezd	6:25	6:35	6:40
6	2	Zlín,,Dlouhá		|	|	|
7	3	Zlín,,nemocnice		6:29	6:39	|
8	4	Zlín,Příluky		6:31	6:42	|
9	5	Želechovice n.Dřev.,,kříž.		6:34	6:45	|
17		lineroute id		1	2	3
18	Přepravu zajišťuje: ČSAD Vsetín a.s., Ohrada 791, 755 01 Vsetín, provozovna Zlín, www.csadvs.cz, odzl@csadvs.cz					
19						
20					INSERT INTO lineroute (l	
21						
24	INSERT INTO lineroutestoptime (lineroute_id, stop_id, time, stop_order) VALUES (					
25		39530		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
26		39535		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
27		39541		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
28		39583		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
29		40159		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
30		39569		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
31		10250		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
32		10248		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
33		10249		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
34		10251		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
35		23204		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
36		23205		INSERT INTO lineroutestoptim	INSERT INTO	INSERT IN
37						

Obrázek 17: Tvorba dat v programu MS Excel (autor, 2019)

4.3 vyhodnoťte vhodnost modelu pro účely výuky a flexibilitu pro účely rozšiřování o další prvky

Model založený na komunikací pomocí VDV301 má několik výhod, které jej činí vhodným pro výuku. Výhody modelu samotného: Pro provozování modelu nejsou nutná žádná specifická zařízení. Dostatečným vybavením jsou alespoň dva počítače s OS Linux a síťový router. V případě potřeby je možné program překompilovat pro OS Windows, nebo jej provozovat na virtualizovaném stroji s OS Linux na počítači s OS Windows.

Jednoduchý způsob tvorby vstupních dat umožňuje výběr libovolné linky ze systému CIS JŘ. Je tedy možné otestovat různé kombinace počtu zastávek na linkách, délky názvu zastávek, nebo podporu znaků národní abecedy.

Výhody komunikačního protokolu pro výuku Kompletní dokumentace ke komunikačnímu protokolu je oficiálně dostupná na internetu (v německém jazyce). Využití XML značkovacího jazyka pro komunikaci umožňuje snadné porozumění studentů charakteru přenášených dat. Průběh komunikace lze sledovat pomocí programu Wireshark běžně dostupného pod Windows i Linuxem.

V aktuálním stavu a rozsahu naprogramovaných služeb lze bez úprav přidávat pouze zařízení využívající údaje o trase. Tedy hlavně TFT displeje. Teoreticky by také bylo možné přidat hlásič zastávek založený na příkazech pro displej, přesto že norma obsahuje speciální příkazy pro hlásič zastávek. Rozšíření o další prvky, jako jsou označovače cestovních dokladů, je čistě otázkou doplnění dalších procedur do služby *CustomerInformationService*.

5 Závěr

S narůstajícími požadavky na objem přenášených dat pro potřeby OIS (přestupní vazby) přestávají současné průmyslové komunikační rozhraní (RS485, RS232, IBIS) kapacitně postačovat. Zároveň s narůstajícím počtem prvků ve vozidle přímo nabízí všechny sjednotit na jednu společnou sběrnici běžnou na průmyslových i osobních počítačích: Ethernet.

Toto rozhraní umožňuje sloučit kabeláž pro informační, odbavovací i kamerový systém. Kabeláž je levná a běžně dostupná, stejně tak aktivní prvky infrastruktury jako přepínače a směrovače.

Současná situace na trhu informačních systémů ve veřejné hromadné dopravě poměrně ulehčuje výběr jednotného otevřeného protokolu pro potřeby organizátorů veřejné dopravy. Jediný aktuálně známý otevřený síťový protokol pro OIS je IBIS-IP (VDV301). Vzhledem k ovládnutí brněnské aglomerace systémem EPISNET[1] bude velmi zajímavé v nejbližších letech sledovat, pro který protokol se rozhodnou organizátoři integrovaných dopravních systémů.

Vzhledem k několika chybějícím funkcionalitám mohou vzniknout obavy o roztříštěnost standardu, pokud budou nějakým subjektem dospecifikovány. V Německu plní roli koordinátora samotný tvůrce normy, spolek VDV. Všechny navržené změny by tedy měly být nejdřív konzultovány s VDV a pokud bude takovýto postup neudržitelný, je

CDV ve spolupráci s ROPIDem ochotné převzít tuto roli koordinátora standardu ve specifikaci pro české provozy.

Nejdůležitější ovšem je, že se standard dostane z rukou výrobců OIS do rukou organizátorů dopravy a tím bude omezena závislost celého systému na jednom dodavateli (tzv. vendor lock).

Co se týká samotného protokolu, přes dostupnost veškeré dokumentace k IBIS IP není implementace této normy do praxe úplně jednoznačným postupem. Norma neobsahuje příklady, na kterých by si vývojář mohl ověřit, zda jsou jeho postupy správné. Chybí veřejný *primer*, tedy stručný výtah tyto nedostatky pokrývající.

Stejně jako v případě rozhraní IBIS nelze dosáhnout stavu "plug and play" bez dodatečné konkretizace požadavků, neboť norma poskytuje příliš mnoho volitelných částí. Specifikace „podpora VDV301“ není sama o sobě dostačující pro funkčnost systému. I přes tyto nevýhody se tento způsob komunikace mezi komponenty vozidlového informačního systému jeví jako velmi konkurenceschopný pro reálný provoz i pro účely výuky. Pokud organizátor dopravy správně dospecifikuje další technické požadavky , jako je například způsob přidělování ip adres v síti, mohou jednotlivé komponenty fungovat v *plug-n-play* režimu nezávisle na dodavateli systému. Struktura dat ve formě XML navíc zaručuje zpětnou kompatibilitu. Přidání dalších vnořených tagů nijak neomezí funkčnost již běžících systémů, které s těmito novými tagy původně nepočítaly.[**?**]

6 Literatura

Reference

[1] **Herman, Ivo:***ŘÍDICÍ INFORMAČNÍ SYSTÉM MHD – RIS II*
Přístup z internetu
`http://www.herman.cz/cs/ridici-informacni-system-vozidla-mhd-ris-ii/`

[2] **Lazar, Guillaume, Penea, Robin:***Storing your data in a database*
Mastering Qt 5

[3] **Brynda, František:***Odbavovací a informační zařízení ve vozidlech PID*
Přístup z internetu
`https://pid.cz/wp-content/uploads/2018/04/Priloha_10_OIS_ve_vozidlech_PID_vcetne_navaznych_priloh.pdf`

7 Seznam obrázků a tabulek

Seznam obrázků

Seznam tabulek

8 Seznam použitých zkratek

API	Application programming interface
ARM	Advanced RISC Machine
ASW	aplikační SW
AVL	Automatic vehicle location
CDV	Centrum dopravního výzkumu
CIS JŘ	Celostátní informační systém o jízdních řádech
ČVUT	České vysoké učení technické v Praze
DB	databáze
DNS	Domain Name System
DNS-SD	Domain Name System – Service Discovery
DOM	Document Object Model
DPMB a.s.	Dopravní podnik města Brna, a.s.
DPO a.s.	Dopravní podnik Ostrava a.s.
DSZO s.r.o.	Dopravní společnost Zlín-Otrokovice, s.r.o.
FD	Fakulta dopravní
FTP	foil screened twisted pair
GNSS	Global Navigation Satellite System
HTTP	Hypertext Transfer Protocol
HW	hardware
ID	identifikátor
IP	internet protocol
JDF	jednotný datový formát
Kordis a.s.	organizátor IDS Jihomoravského kraje
LCD	liquid-crystal display
LED	light emitting diode
mDNS	Multicast DNS
MHD	městská hromadná doprava
MPV	Monitorování provozu vozidel
OBU	On-Board Unit
OIS	Odbavovací a informační systém
PC	personal computer
PID	Pražská integrovaná doprava
PMDP	Plzeňské městské dopravní podniky a.s.
ROPID	Regionální organizátor Pražské integrované dopravy
SQL	Structured Query Language
STP	Shielded twisted pair
SW	software
TCP/IP	Transmission Control Protocol/Internet Protocol
TFT-LCD	Thin-film-transistor liquid-crystal display
UDP	User Datagram Protocol
UI	user interface
VDV	Verband Deutscher Verkehrsunternehmen
XML	eXtensible Markup Language
XSD	XML Schema Definition

9 Seznam příloh - CD s následujícím obsahem:

Složka zdrojové kódy Obsahuje kompletní zdrojové kódy obou programů, lze zkompilovat pomocí QtCreator 4.9.2. pod OS Linux i Windows.

eichlada.sql Nezbytná záloha obsahující všechna vstupní data. Pro funkčnost hlavního programu je potřeba obnovit tuto zálohu do MySQL serveru pomocí PHPmyAdmin.