České vysoké učení technické v Praze
Masarykův ústav vyšších studií
a
Vysoká škola ekonomická v Praze

Podnikání a komerční inženýrství v průmyslu

Bc. Jan Vulc

Využití dat z telematických jednotek osobních automobilů

Diplomová práce

Praha 2016

Vedoucí diplomové práce: RNDr. Tomáš Vaníček, PhD.

Oponent diplomové práce:

Datum obhajoby:

Hodnocení:

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ

V PRAZE

MASARYKŮV ÚSTAV VYŠŠÍCH STUDIÍ

a

VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE

Zadání diplomové práce

Školní rok: 2014/2015

Jméno a příjmení: Bc. Jan Vulc

Studijní program: Podnikání a komerční inženýrství v průmyslu

Obor studia: Podnikání a management v průmyslu

Forma studia: kombinovaná

Téma práce: Využití dat z telematických jednotek osobních automobilů

Téma práce v anglickém jazyce: Use of data from car telematic units

Zásady pro vypracování práce

Cíl práce (stručné vymezení zkoumaného problému): Návrh využití dat z datové sběrnice osobního automobilu a vytvoření modelu aplikace pro jejich vizualizaci.

Teoretická východiska: Telematické systémy umožňují řidiči získat informace o lokaci, pohybu a stavu jejich vozidla. Umožňují také vozidlům bezdrátově komunikovat s jinými vozidly, mobilními telefony, nebo jakýmkoli jiným zařízením připojeným k internetu, což otevírá mnoho dalších možností k práci s těmito daty. K zabezpečení těchto telematických služeb je do systému zapojeno více různých modulů: GPS, WiFi router, audio-video zařízení, bezdrátový komunikační modul či navigační modul. Podle studií existují čtyři stupně vývoje telematiky:

Telematics 1.0: Hands-free hovory a navigace na displeji ve vozidle

Telematics 2.0: Přenosný navigační systém a satelitní rádio

Telematics 3.0: Konektivita (Připojení USB disku, spárování mobilního telefonu)

Telematics 4.0: Bezproblémová komunikace se světem internetu

V práci budou analyzovány moderní systémy spadající do čtvrtého stupně vývoje telematiky. U těchto systémů se předpokládá schopnost zpřístupnit některá data z datových elektronických sběrnic. V teoretické části budou tato data analyzována a bude zkoumán původ informací v nich obsažených. Podle získaných výsledků vznikne v praktické části několik teoretických možností, jak získaná data využít a vytvořit softwarovou aplikaci na práci s nimi.

Metody práce: Pomocí systémové analýzy budou zkoumány dostupné primární a sekundární zdroje zaměřené na problematiku využití dat z vozové sběrnice v telematických systémech. V závěrečné kapitole bude využito metody modelování k vytvoření modelu pro tvorbu telematické aplikace.

Rámcová osnova:

1. Úvod
2. Teoretické vymezení elektroniky osobních vozů, sběrnice CAN a data
3. Návrhy využití dat

4. Analýza modelového případu
5. Tvorba modelu softwarového řešení
6. Závěr

Základní odborná literatura:

1. ARLOW, Jim a Ila NEUSTADT. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. Vyd. 1. Překlad Bogdan Kiszka. Brno: Computer Press, 2007, 567 s. ISBN 978-80-251-1503-9.

2. KANISOVÁ, Hana a Miroslav MÜLLER. UML srozumitelně. 2. aktualiz. vyd. Brno: Computer Press, 2006, 176 s. ISBN 80-251-1083-4.

3. LINDE, Arvid. How Your Car Works: Your Guide to the Components & Systems of Modern Cars, Including Hybrid & Electric Vehicles. Vyd. 1. Veloce Publishing Ltd, 2011, 128 s. ISBN 9781845845001.

4. Ribbens, William B. Understanding automotive electronics.Vyd. 3. Elsevier Science, 2003, 470 s. ISBN 0-7506-7599-3.

5. Grant, August E. a kol. Communications Technology Update and Fundamentals Vyd. 1. CRC Press, 2014, 320 s. ISBN 9781317907916.

Vedoucí práce: RNDr. Tomas Vanicek, PhD.

Podpis vedoucího práce:

Datum odevzdání zadání:

Datum odevzdání práce:

Podpis studenta stvrzující přijetí zadání práce:

Toto zadání platí tři po sobě jdoucí semestry od data odevzdání zadání.

Schválení zadání DP

Datum a podpis vedoucího programu

Podpis ředitele MÚVS

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a že jsem uvedl všechny použité informační zdroje. Zároveň prohlašuji, že veškeré informace týkající se společnosti ŠKODA AUTO a.s. a jejich produktů, použité v této diplomové práci, jsou volně dostupné z internetu, nebo z volně stažitelných aplikací od společnosti ŠKODA AUTO a.s. pro chytré mobilní telefony. V této práci jsem nevyužil žádné interní zdroje, nebo utajené skutečnosti, kterých jsem nabyl během své práce pro technický vývoj ŠKODA AUTO a.s. nebo e4t – electronics for transportation s.r.o.

V Praze, 6. 1. 2016

……..

podpis diplomanta

Děkuji svému vedoucímu práce RNDr. Tomáši Vaníčkovi PhD. za rady, které mi poskytl a za časovou flexibilitu při dokončování této diplomové práce.

Identifikační záznam

Bc. Jan Vulc. *Využití dat z telematických jednotek osobních automobilů.* Praha, 2016. 72 stran, 0 stran příloh. Diplomová práce. České vysoké učení technické v Praze, Masarykův ústav vyšších studií a Vysoká škola ekonomická v Praze, Podnikání a komerční inženýrství v průmyslu. RNDr. Tomáš Vaníček, PhD.

Abstrakt

Tato diplomová práce se věnuje návrhu využití a zpracování telematických dat pro aplikaci elektronické knihy jízd, která by získávala data z datové sběrnice automobilu a z mobilního telefonu řidiče. V úvodu práce jsou vysvětleny základní pojmy z oboru současných elektronických systémů osobních automobilů, jako jsou sběrnice CAN a LIN. Dále jsou představeny současné i budoucí telematické systémy pro sledování vozidel nebo další rozšíření funkcionality auta. Závěrečná část osvětluje zásady jazyka UML, které jsou následně využity při tvorbě modelů mobilní aplikace elektronické knihy jízd.

Abstrakt v anglickém jazyce

This Diploma thesis is about usage of data from car telematics units in a project of smartphone app, which should be able to gather information from car CAN-bus network and driver's mobile phone. There is a short overview on field of electronic systems of today's cars such as CAN-bus network, LIN network etc. Telematics systems to track vehicles or with other functionalities are introduced in the following chapter, including those systems which are not in use yet. Last part describes basics of UML modelling language and a model of smartphone app for electronic driver's logbook is created.

Klíčová slova

Elektronické systémy auta, telematika, data, CAN, UML, model

Klíčová slova v anglickém jazyce

Electronics car systems, telematics, data, CAN, UML, model

Obsah

Předmluva

Téma této diplomové práce spojuje dvě oblasti, kterým se věnuji. Aktuálně pracuji v oblasti vývoje zábavní elektroniky osobních vozů. V minulosti jsem měl na starosti vývoj zasíťování a komunikace elektronických řídících jednotek ve vozidle, takže téma komunikace a přenosu dat na vozidlových sběrnicích je mi blízké a rád navrhuji nové využití pro data, která na sběrnicích jsou.

Blíží se doba, kdy budou telematické jednotky ve vozidlech povinností. Stejně tak se stále více a více sbližují světy automobilů a mobilních telefonů. Již v tuto chvíli umožňuje většina evropských automobilek nějakou formu komunikace mezi vozidlem a chytrým mobilním telefonem cestujících.

V této práci navrhuji využití a zpracování telematických dat pro aplikaci elektronické knihy jízd, která by získávala data z datové sběrnice automobilu a z mobilního telefonu řidiče.

V době vzniku této práce není na trhu dostupná žádná totožná mobilní aplikace, která by mohla nabídnout stejnou funkcionalitu.

1. Úvod

Cílem práce je navrhnout zpracování a využití dat z datové sběrnice osobního automobilu a vytvoření modelu aplikace, která tato data využívá.

Výchozí hypotézou práce je tvrzení, že pokud existují v osobním vozidle elektronické jednotky, které poskytují informace z datových sběrnic volně mimo tyto sběrnice, dají se tato data využít k vylepšení již existujících řešení telematických systémů.

První část práce (kapitola 2) podává informace o zasazení navrhovaného systému do oblasti elektroniky osobního vozu. Jsou zde vysvětleny základní principy nutné k pochopení toho, jak celý elektronický systém a komunikace v něm funguje. V další části (kapitoly 3 a 4) jsou vysvětleny východiska, která vedou k tomu, proč byl pro praktickou část zvolen právě model elektronické knihy jízd využívající data z vozidlových datových sběrnic zprostředkovaných přes jednotku SmartGate ve vozech Škoda Auto. V poslední části (kapitola 5) jsou vysvětleny základní principy modelování systémů v jazyce UML a zároveň jsou aplikovány v praxi při tvorbě modelů navrhované aplikace.

Ke zpracování práce je dostupné množství bibliografických zdrojů. Nejvýznamnější je kniha UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky od J. Arlowa a I Nestadta. Dalším důležitým zdrojem je specifikace sběrnice CAN od firmy Bosch. Kromě bibliografických zdrojů bude využito osobních znalostí a vědomostí z oblasti elektrotechniky osobních automobilů a z oblasti objektového návrhu softwaru.

Práce má celkem 72 stran. Obrázky jsou vloženy v textu tam, kde se na ně odkazuje. Úplný soupis použité literatury je uveden na konci hlavního textu před přílohami, v kapitole Seznam použitých zdrojů. Tento seznam obsahuje všechny prameny, které jsem k této práci využil. Seznam je řazen dle průběžného číslování, tak jak jsou prameny za sebou uvedeny v textu. V hlavním textu na jednotlivé zdroje odkazuji pomocí čísla v hranatých závorkách dle 3. vydání normy ČSN ISO 690.

2. Teoretické vymezení elektroniky osobních vozů, sběrnice CAN a data

2.1. Úvod do elektroniky osobních vozů

Elektronika hraje v automobilu stále významnější roli. V automobilech z počátku devadesátých let bylo pouze pár elektronických jednotek. Kromě ABS ještě motorová jednotka, která řídila vstřikování nebo zapalování motoru.

Jak se postupně zvyšovaly nároky na automobily ze strany zákazníků, regulačních úřadů a ekologických organizací, komplexita elektronického systému v automobilech se zvyšovala. U Volkswagenu Phaeton (facelift 2011) je řídících jednotek 61 a délka kabelů je přes 3800 m.

V současné době umožňuje elektronika splnit tyto základní cíle vývoje vozidla:

- Zvýšení bezpečnosti
- Zvýšení jízdního pohodlí
- Zlepšení životního prostředí
- Zvýšení hospodárnosti

Potenciál původních hydraulických nebo pneumatických pomocných systémů došel jisté technologické nasycenosti, ale ve spojení s mikroelektronikou lze docílit další funkční zlepšení.

Systémy, které redukují jak psychické, tak fyzické zatížení řidiče, snižují únavu řidiče a zlepšují jeho koncentraci, jsou přínosem ke zvýšení aktivní bezpečnosti provozu motorových vozidel.

Elektroniku v osobních automobilech lze rozdělit na čtyři hlavní oblasti:

- Hnací ústrojí
- Komunikace a zábava
- Komfort
- Bezpečnost

Začátek použití elektroniky v motorových vozidlech se datuje na konec 70. let, kdy se začal ve vozech objevovat systém ABS. Tento protiblokovací systém se stal nejdůležitějším příspěvkem k aktivní bezpečnosti. Prvky tohoto systému se používají i pro systém ASR, který zajišťuje zlepšenou stabilitu a trakci.

Další elektronické řídicí systémy zajišťují optimální jízdní provoz např. elektronické řízení převodovky, regulace výkonu a chování motoru, regulace vlastností podvozku atd.

Vykonávání těchto pokročilých úloh vyžaduje výkonné systémy, senzory a komponenty, které rychle a spolehlivě fungují. V současnosti implementované funkce jsou zpravidla založené na spojení několika elektronických jednotek. Například funkce ACC (Adaptive Cruise Control) udržuje nastavenou rychlost, kterou reguluje podle aktuální dopravní situace a požadované vzdálenosti od vozidla před sebou. Řidič si nastaví pouze rychlost a rozestup a dále se o brždění ani zrychlování nestará. Jednotka ACC je zodpovědná za rychlé zpracování dat z předního radaru nebo kamery. Podle potřeby komunikuje s jednotkou brzd a s motorovou jednotkou, případně s automatickou převodovkou. Řidič provádí nastavení požadované rychlosti a rozestupu pomocí tlačítek na multifunkčním volantu nebo pomocí páčky pod volantem a displeje ve sdruženém přístroji. V tomto jednoduchém případě je do spolupráce zapojeno sedm elektronických jednotek a to je pouze část tohoto komplexního systému.

V případě, že je detekován nevyhnutelný střet, je zpravidla spuštěna série procesů nazývaná Pre-crash, která připraví vozidlo na střet. Dnešní vozidla dokonce umí rozpoznat několik druhů střetů a podle toho se zachovat. Mezi standardní procedury ale patří panické brždění, napřímení sedaček, přitažení bezpečnostních pásů, příprava airbagů, uzavření oken, rozblikání varovných světel. Systém vozidla vše načasuje ve správném sledu na správný čas podle údajů o zrychlení, brzdné síle, obsazenosti sedadel, charakteru překážky a očekávaném okamžiku nárazu. Účelem těchto systémů je snaha o zlepšení jízdní bezpečnosti.

Pro umožnění komunikace elektronických jednotek mezi sebou je nutné tyto jednotky propojit. Při konvenčním propojení všech těchto prvků by kabelové svazky byly velmi objemné a rozsáhlé, délka kabelů by byla několik kilometrů. Proto se ve všech moderních automobilech prosadila sběrnicová multiplexní koncepce.

2.2. Datové sběrnice

Stále se zvyšující nutnost výměny informací mezi řídícími jednotkami má pro celkový systém vozidlové elektroniky nesmírný význam. Aby elektrická a elektronická

část zůstala i přesto přehledná, je nutné uplatnit komplexní systém. Jedním z těchto systémů je sběrnice CAN. Ta byla vyvinuta speciálně pro automobilový průmysl a momentálně je využíván ve většině světových automobilek.

Mezi hlavní výhody datové sběrnice patří:

- Zmenšení a zjednodušení kabelových svazků.
- Menší řídící jednotky díky menšímu počtu propojovacích pinů na konektoru. Zmenšení zástavbového prostoru.
- Odpadá využití několika čidel se stejnou funkčností. Více řídících jednotek může používat data z jednoho zdroje.
- Při rozšíření datového protokolu a nové informace není nutné měnit hardware, ale jedná se pouze o softwarovou změnu.
- Díky neustálému ověřování vysílané informace se snížila míra chybovosti.
- Umožnění vysokorychlostního přenosu mezi řídícími jednotkami.
- Usnadněna výměna dat mezi řídícími jednotkami od různých výrobců, protože CAN je mezinárodně standardizovaná sběrnice.

2.2.1. Sběrnice CAN

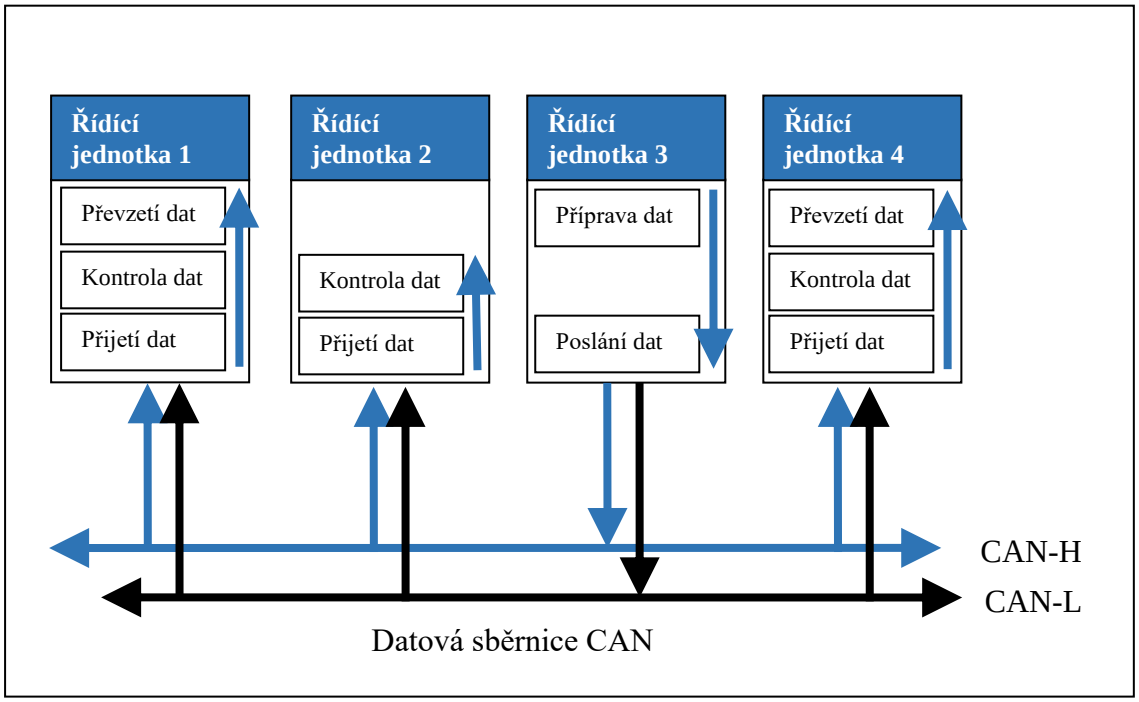
Sběrnice CAN (Controller Area Network) je standardem od roku 1994. Jedná se o sériový komunikační protokol vyvinutý firmou BOSCH v roce 1983 pro nasazení v automobilech, který umožňuje distribuované řízení systémů v reálném čase s velmi vysokou mírou zabezpečení proti chybám. [1]

Tuto sběrnici je možné využít jak v rychlých sítích, tak i v cenově levnějších multiplexových systémech. S Přenosovou rychlostí až k 1Mbit/s najde využití v elektronice v automobilech nebo u ostatních motorových jednotek a senzorů. [2]

CAN je protokol typu multi-master, kde každý node (uzel) na sběrnici se může stát masterem a řídit tak další nody (uzly). Centralizované řízení z jednoho nadřazeného uzlu tedy není nutné, což zjednodušuje řízení a zvyšuje spolehlivost (při poruše jednoho uzlu může zbytek sítě dále pracovat) [1]

Sběrnice s náhodným přístupem řídí přístup k mediu. Tato sběrnice řeší kolize na základě prioritního rozhodování. Komunikace po sběrnici mezi dvěma uzly funguje na

principu zasílání zpráv. Tyto zprávy neobsahují žádné informace o uzlu, pro který jsou určeny. Jsou přijímány všemi uzly, které jsou připojeny ke sběrnici. Na začátku každé zprávy je identifikátor, který udává význam a prioritu zasílané zprávy. Definice protokolu CAN zajišťuje, že zpráva s vyšší prioritou bude v případě kolize vyslána přednostně. Každý uzel zná identifikátory zpráv, které se ho týkají. [1]



Obr. 1: Zobrazení průběhu datového přenosu

Na obrázku 1 je zobrazen průběh datového přenosu na sběrnici CAN mezi čtyřmi modelovými řídícími jednotkami. Řídící jednotka č. 3 vysílá na sběrnici data. Všechny ostatní jednotky naslouchají. Jednotka č. 2 při kontrole dat zjistí, že tato data nepotřebuje a dále s nimi nepracuje. Jednotky č. 1 a č. 4 data převezmou, zpracují a případně vykonají pokyny, které jsou v datech obsaženy.

Sběrnice CAN a její protokol byly normovány v roce 1993 Mezinárodní organizací pro standardizaci. ISO 11898-1 popisuje CAN protokol, ten pokrývá Data-link vrstvu a fyzickou vrstvu ISO-OSI modelu. Tedy tu část, která se implementuje do hardwaru. Další dvě ISO normy 11898-2 a 11898-3 popisují dvě různé verze fyzické vrstvy. Jedna pro high-speed CAN a druhá pro low-speed CAN. Liší se od sebe hlavně definicí napětí a rychlostí přenosu dat. High-speed CAN umožňuje přenosy do 1Mbit/s a

používá se hlavně v oblasti podvozkových systémů a systémů hnací jednotky. Low-speed CAN má rychlost definovanou na 125kbit/s a používá se hlavně tam, kde nejsou takové požadavky na rychlost.

Jak již bylo předesláno v předcházejícím odstavci, pro zajištění transparentnosti návrhu a flexibility implementace je specifikace sběrnice CAN rozdělena do tří vrstev:

CAN vrstva objektů (object layer)

CAN transportní vrstva (transfer layer)

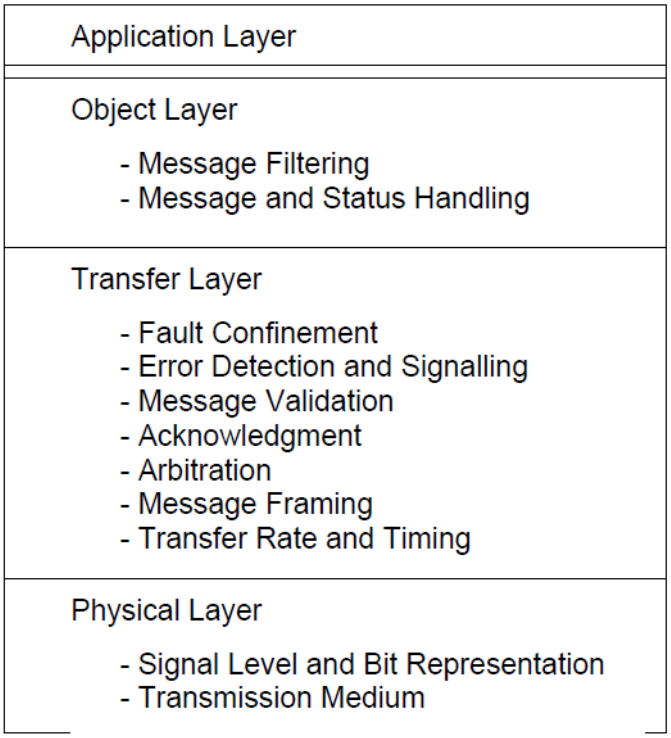
CAN fyzická vrstva (physical layer)

Vrstva objektů a transportní vrstva obsahuje všechny funkce a služby poskytované v rámci linkové vrstvy (data link layer) definované v ISO/OSI modelu. Vrstva objektů má na starosti nalezení zpráv, které mají být odeslány, které zprávy obdržené od transportní vrstvy jsou užitečné a poskytnutí rozhraní pro aplikační vrstvu spojenou s hardwarem. [2]

Vrstva transportní obsluhuje transportní protokol, kontroluje rámce, kontroluje a hlásí chyby na sběrnici a řídí přístup na sběrnici (arbitration). V této vrstvě dochází k rozhodnutí, zda je sběrnice volná pro vysílání, nebo zda je nutné naslouchat. [2]

Fyzická vrstva má na starosti samotný přenos bitů mezi rozdílnými uzly s ohledem na všechny elektrické náležitosti. V rámci jedné sítě musí být fyzická vrstva stejná pro všechny uzly, nicméně její parametry mohou být zvoleny s ohledem na danou aplikaci.[1]

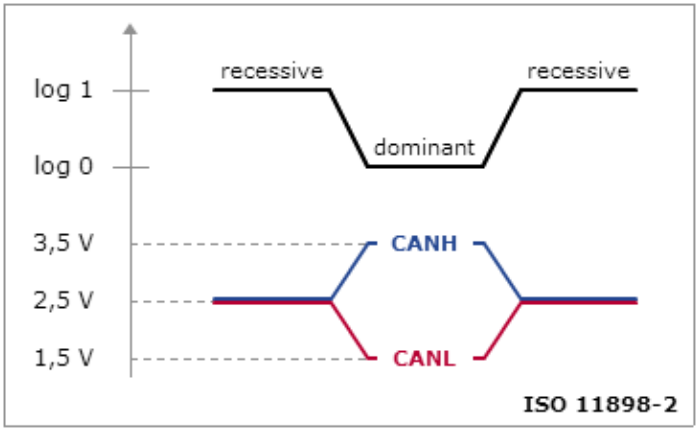
CAN protokol má definované vlastní rozhraní k fyzickému přenosovému mediu, čímž se liší od modelu ISO/OSI známého z jiných oblastí. Tyto vlastnosti fyzické vrstvy jsou ale jeho velkou předností. Základním požadavkem na přenosové medium je, aby mohlo realizovat logický součin. Kvůli zvýšení rychlosti a odolnosti proti rušení musí být spoj symetrický. Specifikace CAN protokolu také definuje dvě vzájemně komplementární hodnoty bitů na sběrnici – dominantní a recesivní. Jedná se v podstatě o ekvivalent logické 1 a logické 0, ale bez určených hodnot. Skutečná reprezentace záleží na konkrétním nastavení fyzické vrstvy v konkrétním případě.[1]



Obr. 2: Struktura vrstev CAN uzlu [2]

Pravidla pro stav na sběrnici jsou jednoduchá. Pokud na sběrnici všechny uzly vysílají recessive bit, pak na sběrnici je úroveň recessive. Pokud alespoň jeden uzel vysílá dominant bit, pak je na sběrnici úroveň dominant. Jednoduše se to dá představit na příkladu optického vlákna, u kterého stav dominant odpovídá tomu, že vlákno svítí a stav recessive nesvítí.[1]

Fyzický přenos dat je založen na přenosu diferenciálních napětí. Tím se efektivně eliminuje negativní efekt interference od motoru, zapalovacích systémů a kontaktů. Transportní medium je složeno ze dvou vodičů CAN high line (CAN-H) a CAN low line (CAN-L). Zakroucením těchto linek je snížen vliv magnetického pole. Specifické hodnoty napětí záleží na použitém CAN rozhraní. Je zde rozdíl mezi high-speed CAN (ISO 11898-2) a low-speed CAN (ISO 11898-3). ISO 11898-2 definuje logickou jedničku jako 0 V, a logickou nulu jako typicky jako 2 V. CAN protokol interpretuje rozdíl napětí mezi vodiči větší než 0,9 V jako dominantní úroveň a hodnotu pod 0,5 V jako recessive úroveň. [3]

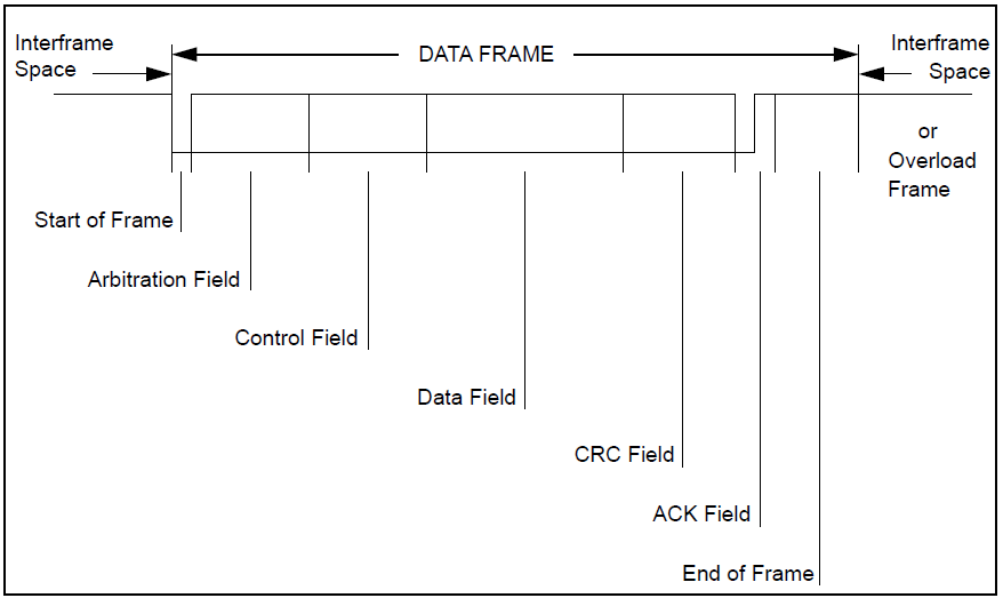


Obr. 3: Úrovně signálu na High-speed CAN vodičích [3]

Informace jsou na sběrnici vysílány ve fixním formátu zpráv o různé délce, která je ale shora omezena. Každá zpráva má pevně danou strukturu.

„V CAN protokolu existují čtyři různé typy rámců:

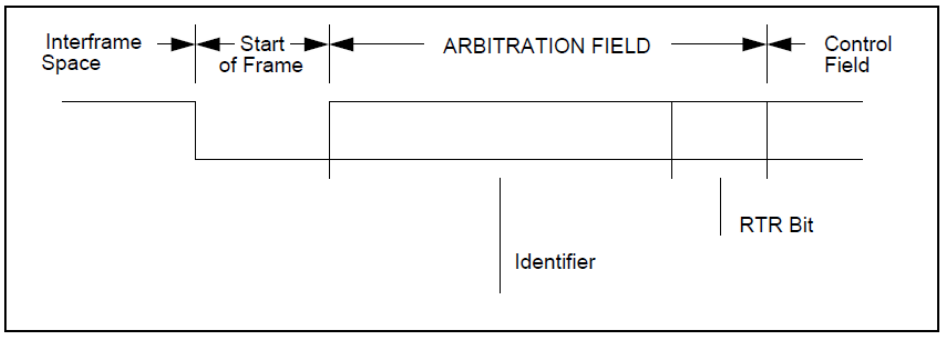
- DATA FRAME – obsahuje data od vysílající jednotky
- REMOTE FRAME – jedná se o žádost o data se stejným identifikátorem
- ERROR FRAME – je vyslán jakoukoli jednotkou v případě detekované chyby na sběrnici
- OVERLOAD FRAME – používá se pro vytvoření mezery mezi jednotlivými datovými rámci.“ [2]



Obr. 4: složení datového rámce [2]

Data Frame, neboli datový rámec je složen ze sedmi různých bitových polí:

- Začátek zprávy = Start Of Frame (SOF) označuje začátek datového rámce. Obsahuje pouze jeden dominantní bit. Pomocí tohoto SOF bitu se všechny jednotky na sběrnici synchronizují.
- Arbitration Field obsahuje identifikátor a RTR-bit. Identifikátor má délku 11 bitů. RTR bit je u datového rámce dominantní u zprávy typu Remote Frame je recesivní.



Obr. 5: složení pole Arbitration v datovém rámci [2]

- Kontrolní pole = Control Field, se skládá ze šesti bitů, kdy 4 jsou vyhrazeny pro indikaci délky dat = Data Length Code. Pomocí 4 bitů je zde indikována délka datového pole 0-8 bytes. Zbylé dva bity jsou rezervovány pro další rozšíření.
- Datové pole = Data Field obsahuje přenášená data. Může obsahovat 0 – 8 bytů
- CRC pole obsahuje zabezpečovací kód a je ukončen recesivním bitem.
- ACK pole, neboli potvrzení obsahuje dva bity. ACK Slot a ACK Delimiter (oddělovač). Při vysílání je ACK slot recesivní, při příjmu jej přijímací jednotka změní na dominantní jako potvrzení, že zpráva je platná a byla přijata korektně. ACK oddělovač je stále nastavený na recesivní hodnotu.
- End Of Frame = konec zprávy obsahuje 7 recesivních bitů. [2]

Remote Frame, neboli žádost o data má podobný formát jako datový rámec. Pouze RTR bit má recesivní hodnotu a nemá žádnou datovou část. V případě, že nějaký z uzlů potřebuje zaslat data, vyšle žádost o data s identifikátorem zprávy, kterou požaduje. Tímto nastavením je zajištěno, že v případě, že nějaký uzel žádá o určitá data a ve stejnou chvíli jiný uzel tato data vysílá, přednost v přístupu na sběrnici má uzel vysílající datovou zprávu, právě kvůli úrovni RTR bitu.[1]

2.2.2. Sběrnice LIN

První specifikace protokolu sběrnice LIN (Local Interconnect Network) byla finalizována v roce 2002. Na jejím vývoji se podílelo pět výrobců automobilů (Volkswagen, BMW, Audi, Volvo, Mercedes-Benz) a dva výrobci elektroniky (Motorola, Volcano). Hlavním úkolem bylo vytvořit levnější variantu sběrnice CAN, která byla pro některé použití zbytečně drahá, rychlá nebo robustní. Nejedná se o náhradu sběrnice CAN, ale o její doplnění u takových jednotek, které nejsou tak náročné na tok dat, rychlost nebo bezpečnost, například elektronika stahování oken, klimatizace, střešních oken atd. [4]

Jedná se o sériovou síť složenou z jednoho mastera a až maximálně 16 slavů. Všechny zprávy jsou inicializovány od mastera a vždy jeden slave odpovídá podle daného identifikátoru zprávy. Protože je veškerá komunikace inicializována od mastera, LIN sběrnice nemá implementovanou žádnou detekci kolize. [4]

Komunikace pomocí sběrnice LIN probíhá pouze po jednom drátu v rychlostech maximálně 19,2 kbit/s při maximální délce vedení 40m. Dle LIN specifikace verze 2.2 je umožněna rychlost až 20 kbit/s.

2.2.3. Příklad využití sběrnic CAN a LIN v osobním vozidle

Přímo připojit všechny jednotky na jednu sběrnici CAN není technicky možné. Pokud je totiž počet aktivních uzlů na sběrnici vysoký, může se snadno stát, že některé jednotky vůbec nedostanou prostor na vysílat, protože bude sběrnice vytížena přenosem zpráv s vyšší prioritou. Zpravidla se uvádí, že vytíženost sběrnice nesmí překročit 80%. Navíc je ekonomické, použít rychlý CAN (High-speed CAN) pouze tam, kde to je opravdu potřeba, tedy u jednotek, které plní bezpečnostní funkci a je třeba, aby rychle

reagovaly na určitou nebezpečnou situaci, např. řídící jednotka airbagů, brzd, motoru atd. Potom jsou jednotky, které přenášejí velká množství dat, například zadní couvací kamera, která za předpokladu, že je obraz ve větším rozlišení a dostatečně kvalitní, by datovou sběrnici celou zahltila. Proto mohou mít sběrnici vlastní. Dále existují jednotky, které nejsou životně relevantní a obstarávají komfort cestujících nebo jejich zábavu. Tyto řídící jednotky byly dříve umísťovány na pomalý CAN (Low-speed CAN), protože použitá přenosová rychlost byla pro tento typ přenášených dat dostatečná. S novými generacemi palubních zábavních a informačních systémů ale toto přestává platit, protože tyto systémy obsahují čím dál více funkcí, které jsou provázané s celým vozidlem, například informace o stavu vozidla, diagnostika a zobrazení chyb na obrazovce palubního počítače, nastavení inteligentních světel, nastavení funkce bezklíčového odemykání, startování a uzamykání, nastavení zvukového zesilovače, personalizace nastavení pro jednotlivé řidiče využívající vůz, ukazatel spotřeby a ekologičnosti jízdy, nastavení nezávislého topení, nastavení jízdních režimů atd. V závislosti na vzrůstající funkcionalitě těchto řídících jednotek vzrůstají také jejich požadavky na využití sběrnice a postupně se dostávají k limitům i rychlého CANu (high-sped CAN).

Proto jsou řídící jednotky rozděleny podle jejich funkčnosti:

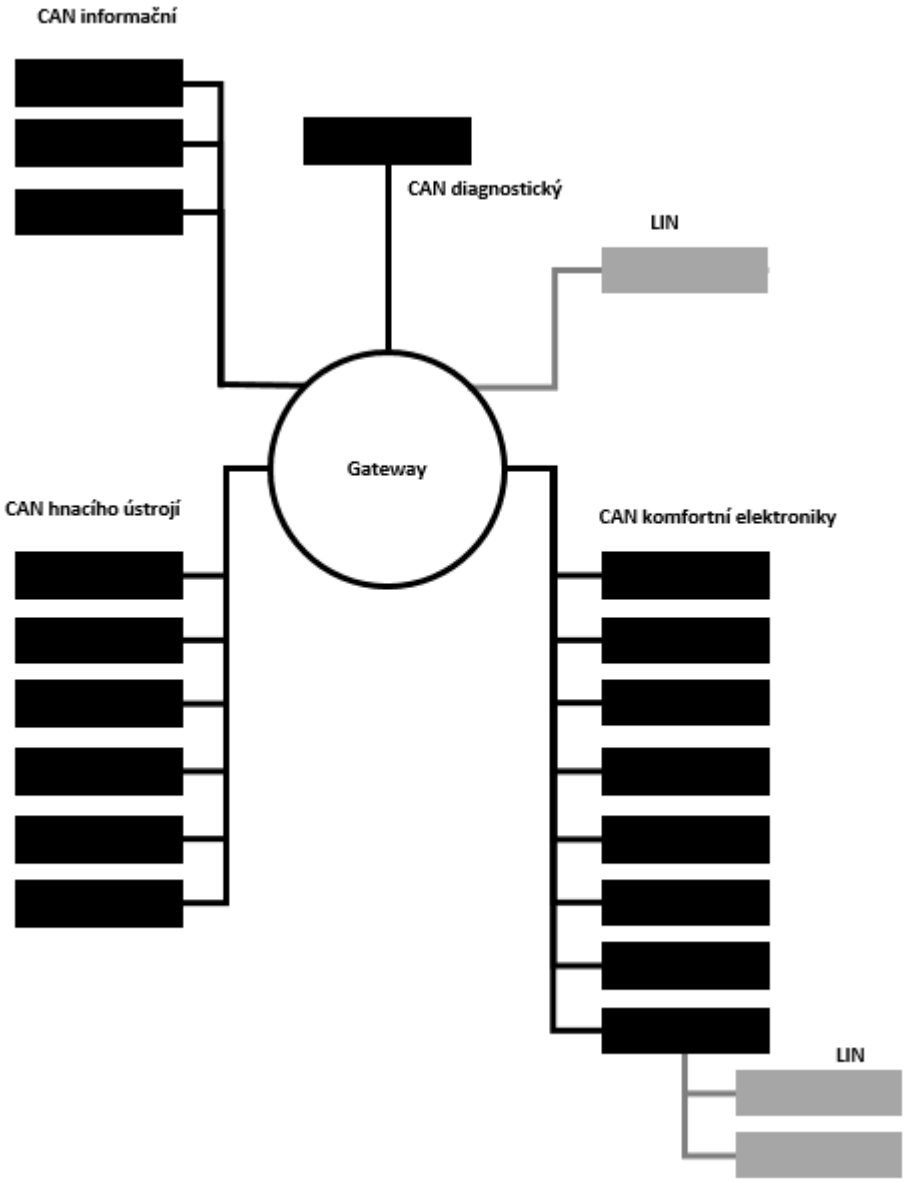
- Hnací ústrojí
- Podvozek
- Bezpečnost
- Komunikace a zábava
- Komfortní elektronika

Každá tato skupina řídících jednotek sdílí jednu sběrnici. Často je ale nutný přenos informací z jedné sběrnice na druhou, například mezi jednotkou multifunkčního volantu, motorovou řídící jednotkou a předním radarem při nastavení adaptivního tempomatu. Přímé propojení těchto datových sběrnic není vždy možné, protože každá z nich může pracovat s různými úrovněmi napětí a může mít jinak uspořádané odpory. Navíc je třeba komunikaci filtrovat a mezi sběrnicemi posílat jen to, co je nutné, aby nedocházelo k zahlcení komunikace. O to se stará jednotka gateway. Může být samostatná, nebo integrovaná např. v jednotce centrální elektroniky, ve sdruženém přístroji atd. Gateway řídí celý provoz na jednotlivých sběrnicích a sleduje jejich

vytíženost. Obsahuje rozsáhlé tabulky všech existujících zpráv a podle nich řídí jejich routování mezi CAN sběrnicemi. Protože CANová datová zpráva obsahuje zpravidla více signálů s rozdílným obsahem, gateway umí vyselektovat potřebné informace z několika zpráv a vytvořit zprávu novou, která je očekávána na další sběrnici. Protože má gateway přístup ke všem sběrnicím, je na ní přímo připojena diagnostická zásuvka, která má svoji privátní sběrnici. Pokud k tomu gateway přes diagnostickou zásuvku OBD dostane příkaz, umí routovat celý obsah zvolené sběrnice na tento diagnostický CAN a usnadní tím diagnostiku rozsáhlejší závady v komunikaci.

Každá jednotka na sběrnici CAN musí mít svoji diagnostiku. Pokud dojde k poruše na nějaké řídící jednotce během provozu vozidla, jsou informace o ní uloženy přímo v té konkrétní jednotce a pomocí diagnostického zařízení je možno informace o chybě načíst. Přeposílání těchto diagnostických zpráv je také inicializováno přes diagnostické zařízení v jednotce gateway.

Nyní je jasné, jak vypadá topologie sítě ve vozidle. Jednotlivé sběrnice jsou zpravidla ve hvězdicovém uspořádání zapojeny do jednotky gateway, jak je patrné na ilustračním obrázku č. 6. Většina jednotek má na privátních vedeních připojeny různé senzory a akční členy. Některé z nich jsou připojeny pomocí vedení LIN



Obr. 6: Modelová topologie sběrnic ve starším voze Škoda

3. Telematika

Telematika je vědní obor, který propojuje informační a telekomunikační technologie s dopravním inženýrstvím a jinými odvětvími. Tímto propojením umožňuje širokou oblast využití. Pomocí telematiky se v dopravě hledají řešení pro optimalizaci dopravy, zvýšení bezpečnosti a snížení nehodovosti, zvýšení komfortu dopravy, zrychlení dopravy, zvýšení kapacity dopravní infrastruktury, zefektivnění hromadné dopravy a její integraci s individuální dopravou, automatizaci dopravních procesů apod.

Obecně lze telematické služby rozdělit do několika oblastí:

- Služby pro cestující a řidiče
- Služby pro provozovatele dopravy a dopravce
- Služby pro správce infrastruktury (správci dopravních cest, správci dopravních uzlů)
- Služby pro veřejnou správu (např. výběr mýtného)
- Služby pro bezpečnostní, záchranný a krizový systém IZS

3.1. Praktické příklady využití telematiky

V následujících příkladech jsou uvedeny využití telematiky spadající hlavně do kategorie služeb pro cestující a řidiče, nebo pro dopravce a provozovatele dopravy. Důvodem je to, že v praktické části se bude práce věnovat právě návrhu systému z těchto kategorií.

3.1.1. Sledování vozidel

Monitorování polohy, pohybu, statusu a chování vozidla, nebo flotily vozidel. Toho je dosáhnuto kombinací GPS přijímače a řídící jednotky ve vozidle. Ve valné většině případů se jedná o balík zahrnující instalaci jednotky do vozidla v servisu, back-end server zpracovávající data z jednotky a front-end aplikaci (většinou jak desktopová, mobilní, tak i webová aplikace), která prezentuje data ze serveru uživateli. Jednotka zpravidla získává data z vozidlových sběrnic a s back-end serverem komunikuje buďto pomocí mobilního internetu, nebo přes sms zprávy s informacemi v přesně definovaném

formátu. Řídící jednotka je zpravidla vybavena SIM kartou, za kterou uživatel platí měsíční paušál.

I pro nejjednodušší systémy tvorby elektronické knihy jízd je nutno vůz vybavit přijímačem GPS signálu a SIM kartou pro datovou komunikaci se systémem.

Pomocí telematických funkcí pro sledování vozidel se dají snadno spravovat flotily osobních a nákladních vozů. Údaje z telematických jednotek mohou usnadnit plánování údržby, správu paliva, optimalizaci tras vozidel atd.

Většina systému na trhu umí rozlišovat soukromé a pracovní jízdy. V režimu soukromé jízdy je možné v podniku aplikovat tzv. skrytý mód, ve kterém nadřízený nevidí, kde se automobil pohyboval, ale zjistí pouze, kolik km najel, případně i další údaje.

Některé další systémy mohou být na přání vybaveny rozlišením uživatele například čipem nebo např. RFID kartou. Tato možnost je například pro dopravní firmy, kterým se střídá více řidičů na jednom autě a firma si chce udržet přehled například pro případ dopravní nehody. Takto získaná data zpravidla slouží pouze pro automatické vyplňování knihy jízd.

3.1.2. Řízení dopravy a Car to X komunikace

Jedná se o využití telematických systémů Car to X, které mají umožňovat obousměrnou komunikaci mezi vozidlem a jiným vozidlem nebo mezi vozidlem a dispečinkem řízení dopravy, nebo mezi vozidlem a nějakým jiným zařízením.

V současnosti je ve fázi testování systém, který umí řidiče informovat o stavu semaforu na nejbližší křižovatce a podle toho navrhnout mírné zpomalení nebo zrychlení, aby vozidlo projelo křižovatkou optimálně na zelenou.

Systémy komunikace vozidel s dispečinkem umožňují řídit individuální i hromadnou dopravu v závislosti na lokální dopravní situaci. Data z vozidel mohou být odesílány do centrálního uzlu, kde mohou navrhnout objízdné trasy, nebo snížit rychlost na nebezpečných úsecích. Kromě toho by se tyto systémy daly využít například k placení mýtného. Daleko dle mého není systém, který bude sledovat, jak řidič dodržuje předpisy a o jejich překročení bude informovat příslušné úřady. Například sledování rychlosti by mohlo fungovat velmi jednoduše.

V současnosti jsou podobné systémy v začátcích. Jediná vozidla, která mají určitou formu komunikace Car to X implementovanou jsou prototypy autonomních vozidel bez řidičů, jejichž testování probíhá již v ostrém provozu, například v Kalifornii.

3.1.3. Sledování návěsů

Jedná se zpravidla o podobné řešení, které může být navíc vybaveno senzory ke sledování obsahu návěsu např. pro zajištění čerstvosti přepravovaných potravin a různými RFID čtečkami pro identifikaci řidiče, překladiště, nebo samotného návěsu. Tyto systémy mohou sloužit také pro komunikaci s celními úřady, které pomocí čtečky mohou snadno zjistit obsah nebo původ zboží, trasu, kterou návěs absolvoval, nebo zkontrolovat dodržování povinných přestávek v jízdě.

3.1.4. Sledování dopravních kontejnerů

Obdobné řešení se stejnou funkcionalitou jako v předchozím příkladu, které může usnadnit plánování využití dopravních kontejnerů, manipulaci s nimi a přesné sledování polohy. Pomocí těchto systémů lze hlídat časy a místa otevření kontejnerů a hlásit je na centrálu.

3.1.5. Rozšíření funkcionality vozu

S jednotkou připojenou na vozidlovou sběrnici CAN se dá získat mnohem více informací, než od samostatně fungující jednotky pro sledování pohybu vozidel. Na základě informací, které se na sběrnici CAN vyskytují, je možné např. sestavit řidičský profil řidiče a zobrazit mu tipy pro ekologickou jízdu nebo pro předcházení krizových situací. Všechny získané informace je možné odesílat na back-end server a zobrazovat je na dispečinku nebo například na mobilním telefonu řidiče a to i zpětně. Pokud se to spojí i s nutností přihlašování řidičů, je možné na dispečinku porovnávat, s jakou jezdí který řidič spotřebou, nebo jak dobře řadí. Dále je možnost sledovat vytíženost jednotlivých aut nebo využití aut jednotlivými řidiči nebo odděleními. Další funkcionalitou je kontrola tankování pohonných hmot. Některé monitorovací systémy

umožňují import dat z tankovacích karet společnosti a porovnání jednotlivých tankování se skutečným množstvím paliva, které v nádrži přibylo. Kromě toho je možné zjistit, jestli byl vůz v dané době skutečně na dané čerpací stanici. [5]

Kromě těchto dohledových funkcí mohou data ze sběrnice CAN být využity pomocí telematické jednotky pro různé zábavní a informační funkce. Například automobilka Škoda Auto představila s příchodem třetí generace vozu Škoda Fabia v roce 2014 řídící jednotku SmartGate. Tato jednotka umí číst ze sběrnic velké množství informací, které se dají vizualizovat v několika mobilních a webových aplikacích živě během jízdy, nebo až po jízdě.

„Rozhraní SmartGate je užitečným nástrojem, který vám otvírá cestu k lepšímu porozumění vašeho vozu. SmartGate dokáže sledovat přibližně 40 různých parametrů ze systémů v automobilu, které lze zobrazit a vyhodnotit prostřednictvím aplikací v chytrých telefonech. Přehled tak můžete mít například o úspornosti, nebo naopak dynamičnosti vašeho jízdního stylu.“[6]

Připojení k jednotce je realizováno pomocí technologie Wi-Fi nebo Wi-Fi Direct, odpadá tedy nutnost mít v jednotce přítomnu SIM kartu. Pro datové přenosy se využívá datové připojení synchronizovaného telefonu. V současné době existuje 7 různých aplikací pro jednotku SmartGate:

- Aplikace Škoda Drive odhalí řidiči ekonomické aspekty jízdy. Řidič má k dispozici přehled o množství spotřebovaného paliva a další informace. Během jízdy se v aplikaci ukazuje efektivita jízdy a ekologické tipy. Informace z aplikace se nahrávají na webový portál a dosažené skóre efektivnosti nebo spotřeby může uživatel snadno sdílet např. na sociální sítě. [6]
- Aplikace Service App nahrazuje palubní literaturu. S touto aplikací má řidič přehled o svém voze kdykoli je potřebuje. [6]
- G-meter ukazuje řidiči aktuální hodnoty přetížení, příčného a podélného zrychlení, otáček motoru, zrychlení, aktuálním zařazeném stupni, aktivace brzd, nebo informace o poloze plynového pedálu. [6]
- MFA Pro je přizpůsobitelný palubní počítač. Umí zobrazit všechny informace z palubního počítače v několika různých typech zobrazení. Kromě standardních kontrolek tlaku oleje, funkčnosti ABS, sepnuté ruční

brzdě, umí zobrazit např. graf rychlosti, otáček atd. Průběh jízdy je možné si nahrávat a zpětně jej zobrazovat, nebo sdílet na sociálních sítích. Řidič si může sám poskládat několik obrazovek, kde si sám určí, kde budou zobrazené jaké ukazatele. [6]

- Performance Pal, shromažďuje všechny ukazatele týkající se dynamičnosti jízdy. Data z aplikace lze také použít pro další analýzu. [6]
- Závodní hra Smart Racer je určena pro spolucestující. Aplikace si bere z vozidlové sběrnice rychlost a podle toho nastavuje rychlost autíčka ve hře. Cílem je projet přes bludiště co nejdále. [6]
- Hra Škoda Little Driver je zaměřena na nejmenší spolucestující. Cílem hry je projet soutěžní okruh, který se v reálném čase generuje podle skutečné jízdy. Dítě má za úkol zatáčet tabletem stejně, jako řidič, ve stejnou dobu brzdit a přidávat plyn, nebo spouštět blinkr při odbočování. Za správně projeté trasy získává body, díky kterým si může vylepšovat svoje auto, nebo nakupovat nová auta do garáže. Kromě toho hra učí děti, jak se chovat bezpečně na silnici.

Obr. 7: Seznam aplikací pro jednotku SmartGate [6]

Obr. 8: Aplikace Little Driver [7]

Podle webových stránek Škoda Auto umí jednotka číst více jak 40 datových signálů [6]. Ne stránkách firma zmiňuje pouze některé. Naštěstí aplikace MFA Pro je dostatečně obsáhlá co se týká signálů, které může zobrazit a na druhou stranu poměrně jednoduchá na nastavení a zobrazení veškerého obsahu. Takže lze snadno zjistit, jaké údaje umí jednotka SmartGate ze sběrnice CAN číst. Je jasné, že to určitě nejsou všechny signály, že některé další signály jsou určeny speciálně pro jiné aplikace. Škoda nechala něco i do dalších aplikací, ale je to nejucelenější seznam, který se dá jednoduše sestavit. Je také velmi pravděpodobné, že data se v jednotce upravují. Surová data na sběrnici nejsou dle mé zkušenosti vhodná pro prezentaci uživatelům. Jsou velmi přesná a poměrně rychle se mění.

Číselné údaje	Kontrolky
Doporučený rychlostní stupeň	ABS
G-přetížení	Airbag
Laterální zrychlení	Sešlápnutí brzdy
Napětí baterie	Dálková světla
Otáčkoměr	Denní svícení
Poloha plynového pedálu	ESP
Počet ujetých kilometrů	Hladina chladicí kapaliny
Počet ujetých kilometrů (cesta)	Hladina oleje
Podélné zrychlení	Pás
Rychlost	Porucha posilovače řízení
Režim automatické převodovky	Potkávací světla
Servisní prohlídka (dny)	Přední mlhovky
Servisní prohlídka (km)	Ruční brzda
Spotřeba paliva (litrů/h)	Teplota chladicí kapaliny
Spotřeba paliva (litrů/100 km)	Tlak oleje
Stav paliva	Tlak pneumatik
VIN kód	Výstraha posilovače řízení
Výměna oleje (dny)	Výstražná světla
Výměna oleje (km)	Zadní mlhovky
Zařazený rychlostní stupeň	Zpáteční světla
Úhel natočení volantu	
Start Stop status	

Tabulka 1: Seznam CAN signálů vysílaných z jednotky SmartGate dle aplikace MFA pro [7]

3.1.6. eCall

Systém eCall je systém, který bude poskytován v rámci jednotného evropského tísňového volání 112. Systém se skládá z řídící jednotky na palubě vozidla, mobilní telekomunikační sítě a center integrovaného záchranného systému 112. [8]

„Palubním systémem eCall využívajícím linku tísňového volání 112 se rozumí nouzový systém sestávající z palubního zařízení a prostředků ke spuštění, řízení a uskutečnění přenosu eCall, který je aktivován buď automaticky prostřednictvím palubních senzorů, nebo manuálně, a který prostřednictvím veřejných sítí mobilních bezdrátových komunikací přenáší minimální soubor údajů a prostřednictvím linky 112

vytváří audio kanál mezi cestujícími ve vozidle a příslušným centrem tísňového volání služby eCall.“ [9]

Podle strategie Evropské komise, měl být tento systém zaveden již od roku 2010, několikrát se však tento termín odsouval. Pravděpodobně kvůli tomu, aby výrobcům vozidel byla poskytnuta dostatečně dlouhá doba, aby se přizpůsobili technickým požadavkům tohoto rozhodnutí. Aktuální termín, od kdy budou muset být nově homologované vozy vybaveny tímto systémem, byl stanoven na 31. března 2018.

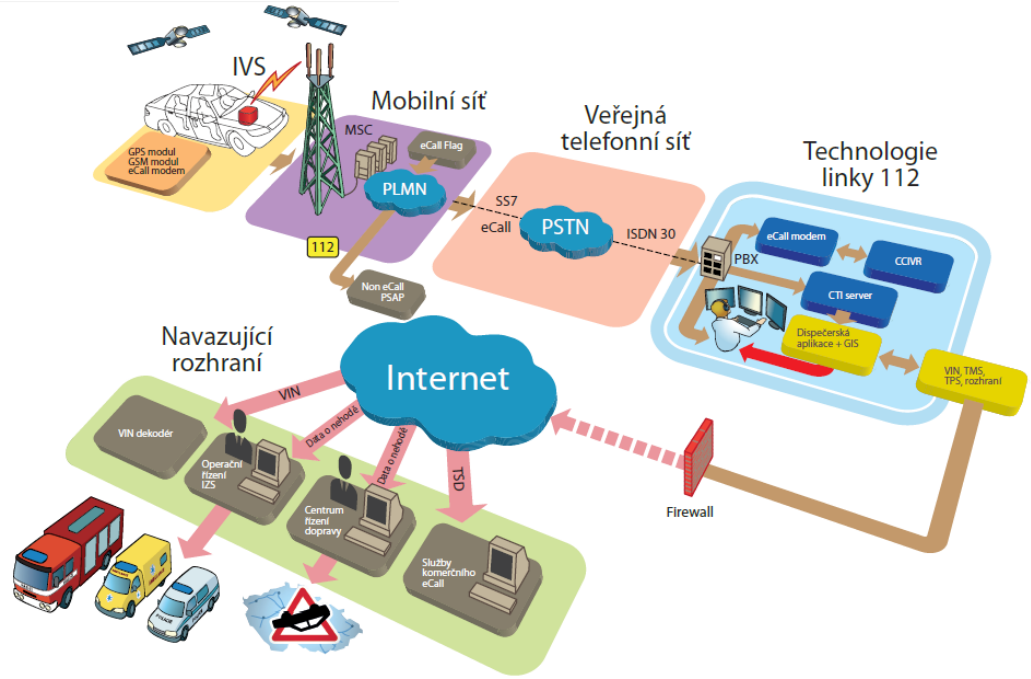
„Od 31. března 2018 bude schválení typu v EU udělováno pouze vozidlům a palubním systémům eCall, které budou v souladu s tímto nařízením.“ [9]

V současné době je v několika evropských zemích zprovozněn komerční systém eCall, kdy tísňové linky neodbavuje call centrum integrovaného záchranného systému, ale smluvně zajištěné asistenční služby, které zajišťují zásah lékařů, odtahových služeb atd. Snahou je tento dočasný projekt nahradit celoevropským systémem, který bude využíván všemi značkami bez ohledu na typ nebo výrobce, zemi registrace nebo polohu automobilu. Navíc s přímou provázaností na bezpečnostní složky. Pouze takto mohou být záchranáři informováni o závažné dopravní nehodě včas i v případě, že ve vozidle vzhledem ke způsobeným zraněním není nikdo schopen komunikovat nebo sám přivolat pomoc pomocí mobilního telefonu.

Od 31. března 2018 bude tedy každé nově homologované osobní auto vybaveno telematickou jednotkou, která bude umístěna na bezpečném místě a připojena na sběrnici CAN. V případě aktivace senzorů umožní hlasové spojení s operátorem na krizové lince 112 a zároveň odešle data, která umožní zasahujícím jednotkám záchranného sboru najít rychle přesné místo nehody a být připraveni na konkrétní situaci. Budou totiž dopředu informováni o počtu pasažérů, podrobnostech o nehodě, typu vozidla, směru jízdy atd. Kromě záchranných složek jsou některé informace předávány i centru pro řízení dopravy. To může okamžitě připravit objízdné trasy, pokud je to potřeba. Obsah a způsob přenosu těchto dat je mezinárodně standardizován pod označení MSD (Minimum set of Data)

Kromě automatického spuštění při nehodě budou moci pasažéři sami ručně spustit tuto službu tlačítkem SOS umístěném na přístrojové desce, nebo u vnitřního zpětného zrcátka.

Přidání další funkcionality, například soukromé asistenční služby nebo dohledových funkcí kvůli prevenci krádeží bude umožněno, pouze pokud s tím zákazník bude souhlasit na základě smluvního ujednání s poskytovatelem služeb, nebo výrobcem automobilu. K tomu bude ale nutné základní jednotku eCall rozšířit o další modul, který tyto funkcionality umožní.



Obr. 9: Architektura řešení eCall v ČR [10]

4. Návrhy využití telematických systémů

Z příkladů uvedených výše je vidět, že telematické systémy se postupně začínají používat ve stále větší míře. Trh jednoduchých systémů pro sledování vozidel ale dle mého názoru již dosáhl nasycení. Správci vozových parků a větších flotil osobních a nákladních vozidel již tyto systémy mají nasazené a dále u nich bude probíhat pouze upgrade na novější generace, ale přechod na kompletně nové, modernější a vyspělejší systémy jsou velkou investicí. Tyto komunikační jednotky se budou pravděpodobně pomalu vyvíjet, dostanou nějakou novou funkcionalitu a budou více komunikovat např. s mobilními telefony, ale daleko zajímavější jsou nové systémy s napojením na vozidlové sběrnice, například systém eCall, který bude plně integrovaný do palubní sítě, a všechna nová vozidla jím budou povinně vybavena. Jak již bylo zmíněno, tak na tomto systému budou moci fungovat i aplikace třetích stran, zatím ale není veřejně

známo, jak bude vozová řídící jednotka přesně fungovat. Povinné uvedení na trh je až 31. 3. 2018. Předpokládám, že možnost přidání aplikací třetích stran přijde ještě později.

Naopak systém SmartGate, představený automobilkou Škoda Auto je již přes rok na trhu a jeho možnosti jsou sice omezené tím, že umí ze sběrnice CAN pouze číst a nikoliv na ni zapisovat, ale i tak může nabídnout velmi širokou škálu využití. V tuto chvíli je dostupných pouze několik aplikací, z nichž většina je přímo od vývojářů Škody, ale pravděpodobně bude možnost dostat se k oficiálnímu SDK. Usuzuji tak z toho, že mezi oficiálně vydanými aplikacemi totiž existuje jedna, která je od firmy Fuerte International. V popisu aplikace na Google play portálu je uvedeno, že jde o výherce soutěže Škoda Hackathon 2014, bohužel internetové stránky této firmy nefungují a ani jiný kontakt nebyl úspěšný. V roce 2014 se jednalo o uzavřenou akci pouze pro zvané vývojáře, kteří zřejmě dostali k dispozici API pro komunikační protokol jednotky SmartGate a na základě toho vytvořili základ aplikace pro mobilní telefony se systémem Android.[11]

V kapitole 3.1.5. Rozšíření funkcionality vozu, je uvedena tabulka základních signálů, které umí řídící jednotka SmartGate číst na vozových sběrnicích a odesílat je pomocí Wi-Fi do mobilního telefonu. Pokud se tyto údaje dají dohromady s informacemi, které má sám o sobě každý chytrý mobilní telefon, např. polohu GPS, nebo údaje o nejbližší buňce mobilní telefonní sítě, dají se využít v celé řadě aplikací.

V modelovém případu v další kapitole bude zpracován model aplikace elektronické knihy jízd využívající data z řídící jednotky SmartGate. Tato aplikace bude vybavena spoustu přídavných funkcí, které současné sledovací systémy a systémy elektronické knihy jízd nemohou bez přístupu k datům z vozové sběrnice zpravidla nabídnout, nebo alespoň ne s takovou přesností.

Touto aplikací by se daly nahradit sledovací jednotky pro flotily vozidel. Navíc nabídne výhody řídící jednotky montované přímo od výrobce, tedy odpadne nutnost dodatečného amatérského zásahu do elektroniky vozidla doma nebo v různých servisech. Majitel vozidla se může v případě potřeby obrátit na kterýkoli autorizovaný servis značky Škoda. Na jednotku je poskytována záruka

Malá nevýhoda je, že přenos z jednotky SmartGate je možný pouze přes Wi-Fi do chytrého mobilního telefonu. Teprve z tohoto telefonu je možný přenos dat do back-end systému a zpřístupnit je dispečerovi, nebo řidiči pro doplnění informací. Naštěstí je

každý chytrý mobilní telefon dnes vybaven mobilním datovým připojením, takže přenos dat do systému může být zajištěn v reálném čase. Další variantou je ukládání dat v paměti mobilního telefonu a jejich přenos až při návratu na základnu, po připojení k podnikové síti.

5. Modelový případ

Pro zpracování modelu aplikace v praktické části se nejvíce hodí jazyk UML. Jedině pomocí tohoto modelovacího jazyka je možné popsat všechny stavy systému a všechny interakce uživatelů se systémem. Kromě toho je možné UML využít pro popis časových souslednosti práce programu.

5.1. Teorie

Jazyk UML (Universal Modeling Language) je univerzální jazyk určený na objektové modelování softwarových systémů a aplikací. Spojuje techniky softwarového inženýrství s modelovacím jazykem. Modely a diagramy vytvořené v jazyku UML mohou být snadno srozumitelné pro lidi i interpretovatelné v programech CASE (Computer-aided software engineering) [12]

Jedná se o souhrn hlavně grafických notací, které slouží k vyjádření analytických a návrhových modelů. Výsledné modely je možné sdílet v rámci vývojového týmu. Mohou sloužit jako zadání pro programátory, nebo jako základ pro vývoj aplikace. Některé modely jsou snadno pochopitelné i pro zadavatele a umožní velmi kvalitní a exaktní vyjasnění všech požadavků na právě vytvářený systém. Pomocí kombinace několika diagramů je možné popsat celý vytvářený systém, protože každý diagram se soustředí vždy na jeden pohled.

„Modelovací jazyk UML je výsledkem snažení analytiků a designérů, kteří v průběhu 80. a 90. let vytvářeli metody, jež by umožnily popsat objektově orientovanou analýzu a návrh. V polovině 90. let byly velmi rozšířené metody OMT (Object modeling Technique), jejímiž autory byli Booch a Rumbaugh a metodika Objectory Ivara Jacobsona. [13]“

První verze sjednoceného jazyka UML přišla v roce 1997 a byla zaštítěna firmou Rational. Postupně se vyvíjejí další verze standardu UML, aktuálně je k dispozici verze 2.5, při tvorbě těchto modelů byla použita notace verze 2.0, protože se aktuálně jedná o celosvětově nejrozšířenější verzi.

Jazyk UML je unifikovaný v mnoha ohledech:

Vývojový cyklus

UML obsahuje vizuální syntaxi pro tvorbu modelu během celého cyklu vývoje softwarového systému. Od požadavků na analýzu od zadavatele a kontrolu během implementace až po závěrečnou kontrolu splnění jednotlivých částí specifikace.

Aplikační doména

Jazyk UML může být využit pro modelování téměř čehokoliv. Od systémů v reálném světě až po podpůrné systémy rozhodování.

Implementační jazyky a platformy

UML je jako jazyk nezávislý na platformě nebo na programovacím jazyku. Samozřejmá je podpora plně objektově orientovaných programovacích jazyků, jako je Java, C#, atd. nebo i pro jazyky založené na objektech, např. Visual Basic. Možnost využít některé typy modelů UML je ale i v neobjektových jazycích jako je třeba C.

Vývojové procesy

Existuje mnoho popsaných metodik vývoje objektově orientovaných systémů. Jazyk UML může podporovat jakoukoli osnovu procesu tvorby softwaru.

5.1.1. Struktura UML

Struktura jazyka UML obsahuje tři součásti. **Stavební bloky** jsou základní prvky modelu. Jako **společné mechanismy** jsou označeny obecné způsoby, jimiž v jazyku UML lze dosáhnout specifických cílů. **Architektura** je pohled v jazyce UML na organizační strukturu a složení navrhovaného systému, včetně jeho rozkladu na součásti, jeho propojitelnost, interakce, mechanismy a zásady, která proniká do návrhu systému. Jedná se o nejvyšší úroveň koncepce systému v jeho vlastním prostředí.

Porozumění struktuře jazyka UML je jedním ze základů navrhování funkčních modelů budoucích systémů.

5.1.2. Objekty

Modely vytvořené dle specifikace UML jsou v podstatě kolekce spolupracujících objektů. Objekty mají jednak statickou strukturu, která popisuje, jaké typy objektů jsou

pro modelování daného systému důležité a jak spolu tyto objekty souvisejí. Naopak dynamické chování objektů popisuje životní cyklus zmíněných objektů a to, jak spolu vzájemně spolupracují, aby dosáhly požadovaných funkcionalit.

5.1.3. Metodika UP

Metodika tvorby softwarového vybavení (SEP – Software Development Process) je jinými slovy proces vývoje softwaru. Tento proces definuje kdo, co, kdy a jak. [12]

„Jedná se o obecnou metodu tvorby softwaru. Pro každou organizaci stejně jako potom pro každý jednotlivý projekt je tedy třeba vytvořit její novou instanci. Tím se uznává, že každý softwarový projekt se od ostatních liší a že model „Tato košile padne všem“ zde rozhodně neplatí. [12]“

Tato metodika UP usiluje o přírůstkovou tvorbu robustní architektury navrhovaného systému [12]. Každý projekt je rozložen na menší projekty a každý z těchto menších projektů je nazýván iterace. Každá iterace může obsahovat těchto pět základních postupů. Nemusí však vždy mít všech pět, závisí to na aktuálním životním cyklu projektu.

Základní pracovní postupy [12]:

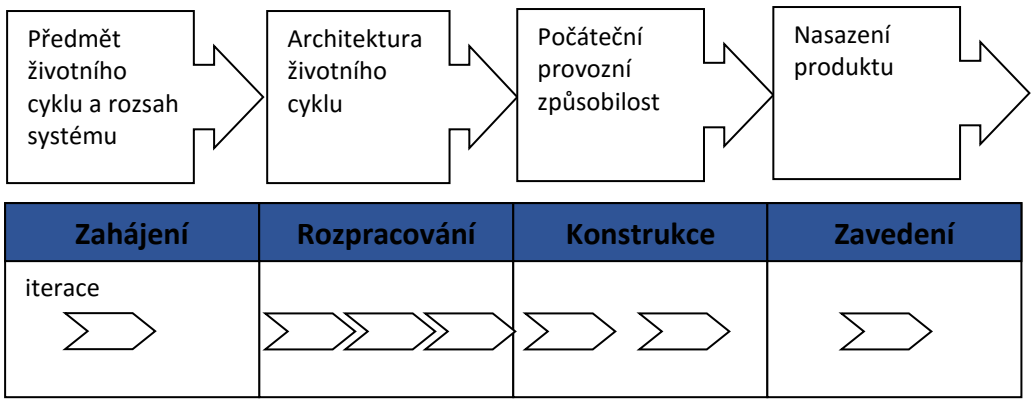
- Požadavky – to, co by měl systém dělat
- Analýza – zpřesnění požadavků a jejich strukturování
- Návrh – Realizace požadavků
- Implementace – Samotná tvorba softwaru
- Testování – Ověření, zda implementace splňuje definované požadavky

Rozložením projektu na několik iterací získáme pružný přístup k plánování projektu. Nejlepší je vytvořit časový plán jednotlivých iterací, kdy často dokončení jedné iterace vede k zahájení další. Nemusí to tak být ale vždy, některé iterace mohou probíhat i současně.

Každá iterace v metodice UP vytvoří základní linii (baseline). Každá baseline poskytuje základ pro další práci a nemůže být jednoduše změněna. Jedině pomocí předem smluvených nástrojů správy změn.

UP má čtyři fáze životního cyklu projektu. Každá fáze má na konci definované hlavní milníky. V každé fázi může proběhnout několik iterací. V každé iteraci bývá

realizováno pět základních pracovních postupů. Počet iterací v každé fázi závisí na velikosti projektu. Každá iterace by neměla být delší než 2-3 měsíce.



Obr. 10: Fáze vývoje dle metodiky UP [12]

Metodika UP se skládá ze čtyř fází vývoje.

- Zahájení
- Rozpracování
- Konstrukce
- Zavedení

Každá fáze vývoje je níže detailněji popsána včetně cílů, kterých musí být na konci každé fáze dosaženo.

Zahájení

Cílem této fáze je odstartovat projekt. Tato fáze obsahuje:

- Tvorba podmínek proveditelnosti – návrh určitých technických prototypů, umožňující ověření správnosti technologických rozhodnutí.
- Nadnesení příkladu, na kterém by mohlo být ukázáno, že projekt bude mít přínos.
- Zachycení všech podstatných požadavků zadavatele, které ovlivní rozsah vznikajícího systému.
- Označení kritických rizik

Konečné cíle neboli milníky je seznam cílů, kterých musí být dosaženo na konci každé fáze.

Hodnotící kritéria (metriky)	Je třeba dodat
Zadavatel souhlasí se záměry životního cyklu	Dokument o představě, jež obsahuje hlavní požadavky na projekt, funkce, a podmínky
Se Zadavatelem byl dohodnut rozsah projektu	Počáteční případ užití (kompletní asi z 10 nebo 20%)
Se zadavatelem byly dohodnuty zachycené klíčové požadavky	Slovník projektu
Zadavatel schválil náklady a pracovní rozvrh	Počáteční plán projektu
Manažer projektu nadnesl obchodní případ	Obchodní případ
Manažer projektu odhadl rizika	Databáze s odhadem rizik
Potvrzení proveditelnosti obsažené v technických studiích a prototypech	Prototypy
Náčrt architektury	Počáteční dokument zachycující architekturu

Tabulka 2: Podmínky splnění milníků první fáze [12]

Rozpracování

Tato fáze obsahuje [12]:

- Tvorbu spustitelného architektonického základu
- Zlepšení odhadu rizik
- Definice atributů kvality
- Zachycení 80% případů užití pro funkční požadavky
- Zpřesnění plánu konstrukční fáze
- Formulace nabídky zahrnující veškeré potřebné prostředky, čas, vybavení, náklady a personál

Hodnotící kritéria (metriky)	Je třeba dodat
Byl vytvořen odolný robustní spustitelný architektonický základ	Spustitelný architektonický základ Statický a dynamický model UML Diagram případů užití
Vize produktu byla stabilizována	Dokument o vizi
Odhad rizik byl revidován	Aktualizovaný odhad rizik
Obchodní případ byl revidován a odsouhlasen zadavatelem	Aktualizovaný obchodní případ
Projekt byl vytvořen do dostatečné hloubky, aby umožnil sestavení realistické nabídky zahrnující odhad času, peněz a prostředků	Aktualizovaný plán projektu
Plán projektu byl porovnán s obchodním případem	Obchodní případ a plán projektu
Bylo dosaženo dohody se zadavatelem o pokračování projektu	Konečný dokument

Tabulka 3: Podmínky splnění milníků druhé fáze [12]

Konstrukce

V této fázi je důležitý pracovní postup implementace. V předchozích fázích byly zachyceny všechny požadavky a vytvořena nabídka a návrh, takže tato fáze je fází výkonnou. Další důležitým úkolem této fáze je testování, ale až ve chvíli, kdy bude tvorba systému natolik pokročilá, že bude potřeba provádět dílčí a větší integrační testy. Stručně je v této fázi nutno dokončit následující [12]:

- Požadavky – nutnost odhalit všechny požadavky, které byly dříve možná přehlédnuty
- Analýza – Dokončení analytického modelu
- Návrh – dokončení modelu návrhu
- Implementace – nutnost zjistit počáteční provozní způsobilost
- Testování počáteční funkční varianty

Hodnotící kritéria (metriky)	Je třeba dodat
Softwarový produkt je dostatečně stabilní a na takové úrovni, aby jej bylo možno nasadit na počítače zadavatele	Produkt UML model Testovací sada
Zadavatel souhlasí s nasazením softwaru do svého prostředí a je k němu připraven	Uživatelské příručky Popis verze
Poměr skutečných výdajů vůči plánovaným je přijatelný	Plán projektu

Tabulka 4: Milníky fáze konstrukce [12]

Fází konstrukce a fází zavedení se tato práce nebude zabývat. Cílem práce je vytvořit funkční model ukazující práci s daty, takže tato a následující fáze je zde uvedena jako doplnění pro úplnost celé této kapitoly.

Zavedení

Zaváděcí fáze začíná, teprve když je dokončeno testování a konečné nasazení systému. Tedy včetně oprav všech chyb z beta-verze a příprava na přenesení systému na všechny zařízení zadavatele. Cíle jsou následující:

- Oprava chyb
- Příprava zadavatelova pracoviště na přijetí nového systému
- Úprava systému v případě vzniku neočekávaných problémů
- Tvorba manuálu a veškeré dokumentace
- Konzultace s uživateli
- Koncová revize

Toto je poslední milník – beta-testy, testy při přejímání a oprava případných chyb jsou dokončeny a produkt je uvolněn k užívání

Hodnotící kritéria (metriky)	Je třeba dodat
Beta-testy jsou dokončeny, byly provedeny nezbytné změny a uživatel souhlasí s faktem, že systém byl úspěšně nasazen	Softwarový produkt
Pracovníci zadavatel produkt aktivně využívají	
Strategie další podpory pro zadavatele byla nejprve dohodnuta a implementována	Plán uživatelské podpory Uživatelské příručky

Tabulka 5: Milníky fáze zavedení [12]

Ve vypracovávaném projektu modelu elektronické knihy jízd pro jednotku SmartGate od firmy Škoda Auto nebudou poslední dvě fáze realizovány. V tuto chvíli se nejedná o reálný projekt, spíše o předvedení jednotlivých metod a modelů, které se dají využít pro návrh aplikace v jazyce UML. Pro samotný vznik aplikace by bylo nutné získat vývojářské API a návod přímo od firmy Škoda Auto, která jej však v současné době volně neposkytuje.

5.2. Identifikace požadavků

Ještě před fází objektové orientované analýzy je třeba si ujasnit, čeho je třeba v projektu dosáhnout a jaký je smysl požadavků a jejich specifikace.

Základem pro požadavky je zpravidla zadávací dokumentace, která však musí být upřesněna. Dalším zdrojem informací jsou interview s uživateli a se zadavatelem. Pro tento krok je však potřeba mít na straně zadavatele kvalitní a kompetentní partnery. Mnoho informací pro formulaci požadavků se dá také zjistit studiem dané domény (oboru). Je nutno prostudovat obvyklá řešení, normy, pojmy a standardy. Požadavky musí také odrážet firemní procesy zadavatele.[13]

Požadavky se dělí na funkční požadavky a nefunkční požadavky. Funkční požadavky je možno definovat v modelu případů užití a popisují požadovanou funkcionalitu systému. Nefunkční požadavky definují proces vývoje nebo omezení, která jsou na systém kladena. Může se jednat o výkonnost, rychlost odezvy, použité standardy, nebo spolehlivost. Nefunkční požadavky však nejsou vhodné na zachycení v diagramu způsobů užití.

Je vhodné uspořádat požadavky do tzv. taxonomie. Jedná se o hierarchii požadavků, kterou lze využít pro jejich kategorizaci. Je to možnost, jak usnadnit práci s velkým počtem požadavků a rozdělit je do menších balíků.[12]

Je navíc vhodné u každého požadavku mít několik atributů. Například datum splnění, jejich zdroj, nebo prioritu. Pokud vyvíjíme aplikaci, která bude mít několik verzí, je možné v jednom z atributů naznačit, kterých verzí se bude daný požadavek týkat. Jedním z důležitých atributů je status. Každý požadavek může být v jednom z následujících stavů:

Proposed (Navržený)	Požadavek nebyl diskutován se všemi stranami a nebyl odsouhlasen
Approved (Schválený)	Požadavek byl oběma stranami odsouhlasen a může být implementován
Rejected (Odmítnutý)	Požadavek byl jednou nebo oběma stranami odmítnut a nebude implementován
Incorporated (Začleněný)	Požadavek byl v nějaké verzi implementován

Tabulka 6: Status požadavků

5.2.1. Identifikace funkčních požadavků pro aplikaci knihy jízd

Pro aplikaci knihy jízd není žádný zadavatel, definice požadavků probíhala podle průzkumů trhu dodatečně montovaných systému pro tvorbu elektronické knihy jízd a na základě komunikace se zástupcem dopravní firmy, která tyto systémy dlouhou dobu využívá. Byl definován okruh funkcionalit, které jsou obvykle od podobných systémů očekávány, a tento základ byl doplněn o mnoho funkcionalit, které umí jednotka SmartGate dodat.

U všech požadavků bylo určeno několik parametrů pro budoucí usnadnění práce s nimi:

- **ID** je pořadové číslo požadavku. Toto označení je unikátní pro každý požadavek. Čísla požadavků nemusí jít popořadě, ale podle toho, jak požadavky přibývají.
- **Typ** rozděluje požadavky do dvou typů. Požadavek typu Nadpis ve skutečnosti požadavkem není, je pouze vložen pro jednodušší rozčlenění do kategorií.
- **Text** je samotný text požadavku nebo nadpisu
- **Status** je u všech požadavků nastaven do hodnoty Approved

Byla vytvořena základní taxonomie, podle které jsou funkční požadavky roztříděné na menší skupinky, tak aby se v nich dalo jednodušeji orientovat:

1.0 Funkční požadavky

- **1.1** Funkční požadavky na připojení
- **1.2** Funkční požadavky na příjem dat
- **1.3** Funkční požadavky na uchování a odesílání dat
- **1.4** Funkční požadavky na uživatelskou interakci
- **1.5** Funkční požadavky na uživatelské role
- **1.6** Funkční požadavky na webové rozhraní

2.0 Nefunkční požadavky

Následující funkční požadavky byly nalezeny pro oblast připojení k jednotce SmartGate. Jedná se v podstatě o to, aby byla aplikace schopna se k jednotce připojit a udržet připojení během celé jízdy.

ID	Typ	Text	Status
1	Nadpis	1.0 Funkční požadavky	Approved
2	Nadpis	1.1 Funkční požadavky na připojení	Approved
3	Požadavek	Aplikace bude při spuštění schopna zkontrolovat připojení k jednotce SmartGate	Approved
4	Požadavek	Aplikace bude schopna připojení k jednotce SmartGate přes Wi-Fi rozhraní	Approved
5	Požadavek	Aplikace bude při spuštění schopna sama zjistit, jestli je na telefonu zapnutý přenos Wi-Fi a případně si vyžádat jeho zapnutí	Approved
41	Požadavek	Připojení k jednotce SmartGate bude umožněno během celého běhu aplikace bez žádných přerušení	Approved

Tabulka 7: Funkční požadavky na aplikaci Kniha jízd v oblasti připojení

ID	Typ	Text	Status
6	Nadpis	1.2 Funkční požadavky na příjem dat	Approved
7	Požadavek	Aplikace bude schopná přijímat předem definovaná data z jednotky SmartGate	Approved
8	Požadavek	Aplikace bude přijímat následující data kontrolek: ABS, Aktivace Airbagu, ESP, Hladina chladicí kapaliny, Hladina oleje, Porucha posilovače řízení, Výstraha posilovače řízení, Teplota chladicí kapaliny, Tlak pneumatik	Approved
9	Požadavek	Aplikace bude přijímat hodnoty těchto proměnných: Počet ujetých kilometrů (celkem), Počet ujetých kilometrů (cesta), Rychlost, Servisní prohlídka (dny a km), výměna oleje (dny a km), Spotřeba paliva, Stav paliva, VIN kód.	Approved
10	Požadavek	Aplikace bude připravena na budoucí rozšíření přijímaných dat za účelem přidání nové funkcionality	Approved
11	Požadavek	Aplikace bude schopná pracovat s GPS daty z mobilního telefonu	Approved

Tabulka 8: Funkční požadavky na aplikaci Kniha jízd v oblasti příjmu dat

Pro příjem dat bylo identifikováno pět požadavků, které jsou napsány v předcházející tabulce č. 8. Tyto požadavky definují, jaká data aplikace přijímá z vozových sběrnic přes jednotku SmartGate a jaké informace získává z mobilního telefonu. Pro snadnější sledování splnění všech požadavků se zadavatelem by bylo vhodnější jednotlivé signály, které aplikace přijímá mít v samostatných požadavcích, pro tento případ, kdy žádný zadavatel není, to není nutné rozepisovat a je to popsáno pouze ve dvou požadavcích č. 8 a 9.

Na oblast uchování a odesílání dat byly definovány 4 základní požadavky. V této části je nutné, aby si aplikace uchovávala všechny data v interní paměti telefonu, dokud nebude schopná provést synchronizaci se serverem. Tato synchronizace může probíhat v reálném čase, nebo v okamžiku, kdy uživatel synchronizaci sám spustí. To, aby synchronizace probíhala v určených intervalech, např. jednou denně je nutné ošetřit v rámci vnitrofiremních předpisů. V aplikaci je nutné také ošetřit to, aby byl uživatel schopen nastavit si, jestli mají být k synchronizaci použity mobilní data, nebo pouze Wi-Fi síť například při návratu na základnu a připojení do vnitropodnikové sítě.

ID	Typ	Text	Status
12	**Nadpis**	**1.3 Funkční požadavky na uchování a odesílání dat**	**Approved**
13	Požadavek	Aplikace bude schopná ukládat přijímaná data do paměti telefonu společně s časovou značkou jejich přijetí	Approved
14	Požadavek	Aplikace bude schopná uložená data synchronizovat se serverem	Approved
15	Požadavek	Synchronizace bude možná v reálném čase, nebo pomocí ručního spuštění uživatelem	Approved
16	Požadavek	Synchronizace bude moci probíhat pomocí mobilního datového přenosu, nebo pomocí Wi-Fi připojení	Approved

Tabulka 9: Funkční požadavky na aplikaci Kniha jízd v oblasti uchování a odesílání dat

V oblasti interakce s uživatelem byly na aplikaci vzneseny následující požadavky:

ID	Typ	Text	Status
17	Nadpis	1.4 Funkční požadavky na uživatelskou interakci	Approved
18	Požadavek	Aplikace bude pomocí GPS navigace v telefonu sledovat uživatelovu polohu a zobrazovat tuto polohu na displeji telefonu v mapě	Approved
19	Požadavek	V případě zastavení vozidla a vypnutí motoru se uživateli zobrazí vyskakovací okno, kde vybere, jestli se jedná o cíl trasy nebo mezizastávku	Approved
20	Požadavek	V případě, že systém detekuje přírůstek na hodnotě množství paliva v nádrži, zobrazí uživateli vyskakovací okno k potvrzení s informací o počtu litrů	Approved
21	Požadavek	Aplikace při prvním spuštění vyžádá přihlášení, přihlašovací údaje v ní zůstanou uloženy	Approved
22	Požadavek	Přihlašovací údaje uživatele budou vždy při otevření aplikace ověřovány na serveru	Approved
23	Požadavek	Uživatel si bude moci nastavit, zda synchronizace se serverem bude probíhat v reálném čase anebo pouze na vyzvání a zda pouze při připojení k podnikové Wi-Fi síti, nebo přes mobilní datový přenos	Approved
24	Požadavek	V případě změny stavu na kontrolkách definovaných v požadavku ID 8, zobrazí se uživateli vyskakovací hláška, která jej bude informovat. Zároveň se tato událost uloží do paměti	Approved

Tabulka 10: Funkční požadavky na aplikaci Kniha jízd v oblasti uživatelské interakce

V předcházející oblasti byly definovány požadavky na uživatelské rozhraní a interakci s uživatelem. Bylo definováno, jaké informace se mají objevit ve formě speciálních vyskakovacích oken vyžadující potvrzení uživatele a také to, že poloha získaná z GPS telefonu se musí zobrazit na mapě na hlavní obrazovce aplikace.

V rámci celého systému musejí být definované uživatelské role. V podstatě lze uživatele rozdělit do tří kategorií. Řidiči pracují denně s aplikací i se systémem přes webové rozhraní. Pomocí aplikace nahrávají data, která mohou ve webovém rozhraní upravovat a hlavně doplňovat o popisky jednotlivých tras a událostí během nich. Četnost doplňování dat a jejich verifikace je opět otázkou vnitropodnikových předpisů,

dá se to ale ošetřit i přímo v aplikaci nebo ve webovém rozhraní. Správci vozů mají možnost nahlížet na všechny jízdy všech vozů a libovolně tato data upravovat. Kromě toho mají na starosti přidávání nových vozů a řidičů do systému. U každého vozu vidí, jaký je zbývající servisní interval a jestli jsou všechny důležité systémy vozidla v pořádku. Do karty vozu se totiž přenášejí všechny rozsvícení kontrolek, které jsou definovány v požadavku ID 8.

ID	Typ	Text	Status
25	**Nadpis**	**1.5 Funkční požadavky na uživatelské role**	**Approved**
26	Požadavek	Budou existovat tři druhy uživatelských rolí: Administrátor, Správce vozů, Řidič	Approved
27	Požadavek	Řidič se bude moci svými přihlašovacími údaji přihlašovat do mobilní aplikace, kde po jeho přihlášení bude moci synchronizovat data získaná z vozu na server. Ve webovém rozhraní má k dispozici pouze údaje ze svých jízd a může je editovat	Approved
28	Požadavek	Správce vozů bude mít všechny oprávnění řidiče a navíc bude moci sledovat všechny jízdy všech aut, přidávat nové vozy, přidávat nové řidiče, sledovat stav aut	Approved
29	Požadavek	Administrátor nebude mít přístup k datům z jízd, bude mít náhled pouze na statistiky jízd a aut. Bude moci nastavovat přístupy, přidávat další administrátory a správce vozů	Approved

Tabulka 11: Funkční požadavky na aplikaci Kniha jízd v oblasti uživatelských rolí

Mnohokrát zmiňované webové rozhraní je definováno pouze základními požadavky. To proto, že v rámci této práce nebude část webového rozhraní celého systému hlouběji zpracovávána. V první fázi projektu je nutné se zaměřit na vývoj a testování samotné aplikace pro sběr dat a jejich přenos do databáze na serveru.

ID	Typ	Text	Status
30	Nadpis	1.6 Funkční požadavky na webové rozhraní	Approved
31	Požadavek	Přes webové rozhraní budou mít uživatelé přístup k datům dle své role	Approved
32	Požadavek	Ve webovém rozhraní bude mít každý řidič svoji kartu, ve které si bude moci zobrazit a editovat svoje jízdy včetně všech informací.	Approved
33	Požadavek	Informace budou ve webovém rozhraní interpretována v několika formách	Approved
34	Požadavek	Každý vůz bude mít ve webovém rozhraní svoji kartu, kde budou všechny jízdy daného vozu nezávisle na řidičích. Správce vozů bude moci tyto údaje editovat	Approved
35	Požadavek	Na úvodní stránce po přihlášení bude mít každý uživatel shrnutí nejdůležitějších pro sebe relevantních informací	Approved

Tabulka 12: Funkční požadavky na aplikaci Kniha jízd pro webové rozhraní

5.2.2. Identifikace nefunkčních požadavků pro aplikaci knihy jízd

Mezi nefunkčními požadavky se objevily požadavky na to, aby aplikace a webové rozhraní byly multiplatformní a na způsob uchování dat na serveru.

Kromě toho se objevily obecné požadavky na GUI webového rozhraní a rychlost odezvy.

ID	Typ	Text	Status
36	Nadpis	2.0 Nefunkční požadavky	Approved
37	Požadavek	Aplikace bude vyvíjena jako multiplatformní, bude spustitelná na systémech iOS a Android	Approved
38	Požadavek	Data budou na serveru synchronizována do databáze MS-SQL	Approved
39	Požadavek	Odezva aplikace bude okamžitá	Approved
40	Požadavek	Aplikace bude mít jednoduché a přehledné GUI v českém jazyce	Approved
42	Požadavek	Webové rozhraní bude spustitelné v internetových prohlížečích nezávisle na operačním systému.	Approved
43	Požadavek	Mezi podporované internetové prohlížeče bude patřit Internet Explorer, Mozilla Firefox, Safari v aktuálních verzích	Approved

Tabulka 13: Nefunkční požadavky na aplikaci Kniha jízd

5.3. Identifikace případů užití

Modely případů užití jsou další formou požadavků a jedná se o jinou formu získání a dokumentace požadavků. Modelování případů užití se skládá z následujících kroků:

- Nalezení hranic systému
- Identifikace aktérů
- Definice případů užití

Výstupem po provedení výše uvedených kroků je model případů užití, který je složen ze čtyř základních komponent:

- Hranice systému – ohraničení kolem případů užití, které zobrazuje hranice modelovaného systému.
- Aktéři – jedná se o role osob nebo předmětů, které systém používají
- Případy užití – jedná se o jednotlivé činnosti, které mohou aktéři se systémem vykonávat
- Relace – vztahy mezi aktéry a případy užití [12]

5.3.1. Nalezení aktérů a případů užití

Nejprve je třeba určit hranice systému. Je to první krok, nad kterým je třeba se zamyslet při tvorbě modelu případů užití. Je totiž třeba určit co je součástí systému a co jeho součástí není. Stanovení hranic systému ovlivňuje funkční požadavky. Dle specifikace UML2 se hranice požadavku označují jako subjekt.

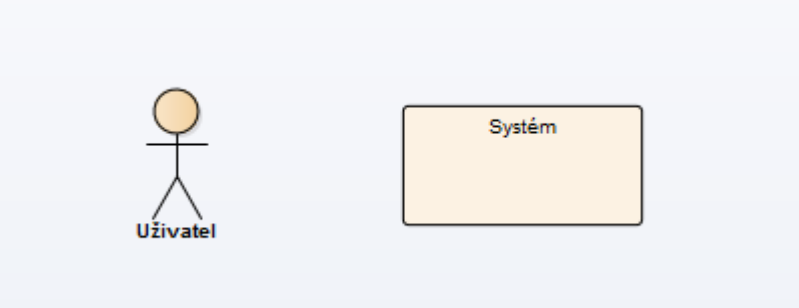
Subjekt je graficky vyjádřen jako rámeček kolem případů užití s popiskem s názvem systému. Aktéři jsou mimo tento rámeček. Během modelování se tento rámeček umístí provizorně a následně se jeho poloha upřesňuje.

Aktér je v podstatě obecná role, kterou může uživatel přijmout v okamžiku, kdy začíná se systémem pracovat. Nemusí se jednat o člověka, ale jako aktér může vystupovat i jiný systém, který může s modelovaným systémem pracovat přes určité rozhraní. Uživatelé nebo systémy mohou nabývat různých rolí, postupně nebo dokonce i

souběžně. V případě, že se v systému stane něco v určitém časovém okamžiku a není to způsobeno žádným aktérem, například pravidelná aktualizace systému, je možné zavést aktéra čas.

Základní chybou při definici aktérů bývá záměna role, která hraje určitou úlohu v systému s konkrétním člověkem nebo předmětem.

Aktéři bývají v UML znázorněni ikonkou panáčka v případě, že se jedná o roli přidělovanou osobám, obdélníkem v případě, že se jedná o roli, kterou hraje nějaký předmět nebo systém.



Obr. 11: různé znázornění aktérů [12]

Při hledání aktérů je nutné zjistit odpovědi na následující otázky:

- Kdo nebo co systém používá?
- Jakou roli ve své interakci hraje?
- Kdo systém instaluje?
- Kdo systém spouští a vypíná?
- Kdo se stará o údržbu systému?
- Kdo do systému zadává data a kdo s nimi pak pracuje?
- Děje se něco v určitou dobu?

Dále je třeba myslet na to, že aktéři jsou vůči systému vždy externí a není možné je kontrolovat. Komunikují se systémem napřímo, mají smysluplný a přesný název a krátký popis.

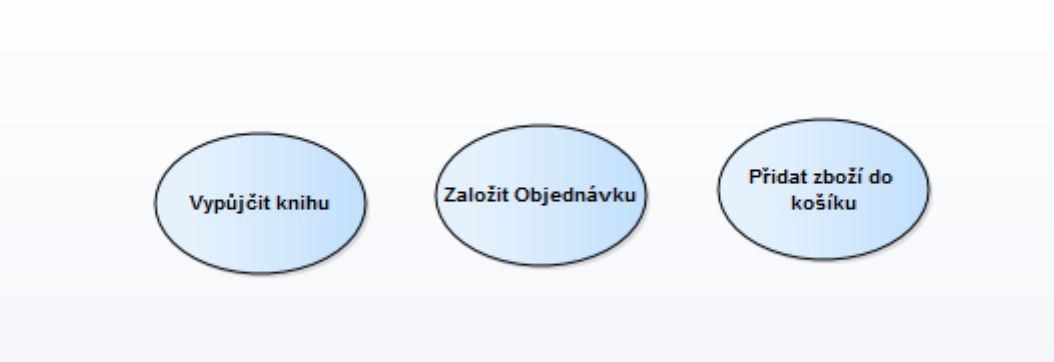
Po definování aktérů a hranic systému je na řadě určení případů užití.

„Případ užití popisuje funkce, které systém poskytuje k užitku jednoho nebo více aktérů. [12]“ Jsou to akce, které od systému aktéři očekávají. Vždy je iniciuje některý aktér a jsou psány z pohledu aktéra. Při definici případů užití je možné najít další aktéry,

to je naprosto v pořádku. Pro usnadnění jejich definice je možné odpovědět na následující otázky [12]:

- Jaké funkce jsou jednotlivými aktéry od systému očekávané?
- Budou v systému uchovávané informace?
- Bude je systém poskytovat?
- Jací aktéři budou tyto činnosti aktivovat?
- Reaguje systém na nějaké vnější systémy?

Graficky případy užití znázorňujeme zpravidla jako ovály s názvem případu užití



Obr. 12: Grafické znázornění případů užití

Každý případ užití má svůj scénář, který popisuje sekvenci kroků, které popisují, jak probíhá interakce mezi systémem a aktérem. V prostředí nástrojů CASE není obecně žádná strukturovaná forma psaní scénářů, proto se standardně používá zápis ve formě tabulky nebo textu a struktura scénáře se zdůrazňuje odsazením nebo vynechanými řádky. [13]

5.4. Diagram případů užití

Pro grafické ztvárnění případů užití a jejich vazeb se využívá Diagram případů užití, tzv. Use-Case diagram.

Při tvorbě diagramů případů užití se rámečkem vyjadřuje subjekt. Aktéry znázorňujeme vně systému, zatímco případy užití jsou uvnitř subjektu. Vztah mezi aktérem a případem užití je vyjádřen plnou čárou, což je dle jazyka UML symbol přiřazení. [14]

Dále je možné definovat tyto základní vztahy:

- Relace <<include>>
- Generalizace
- Relace <<extend>>

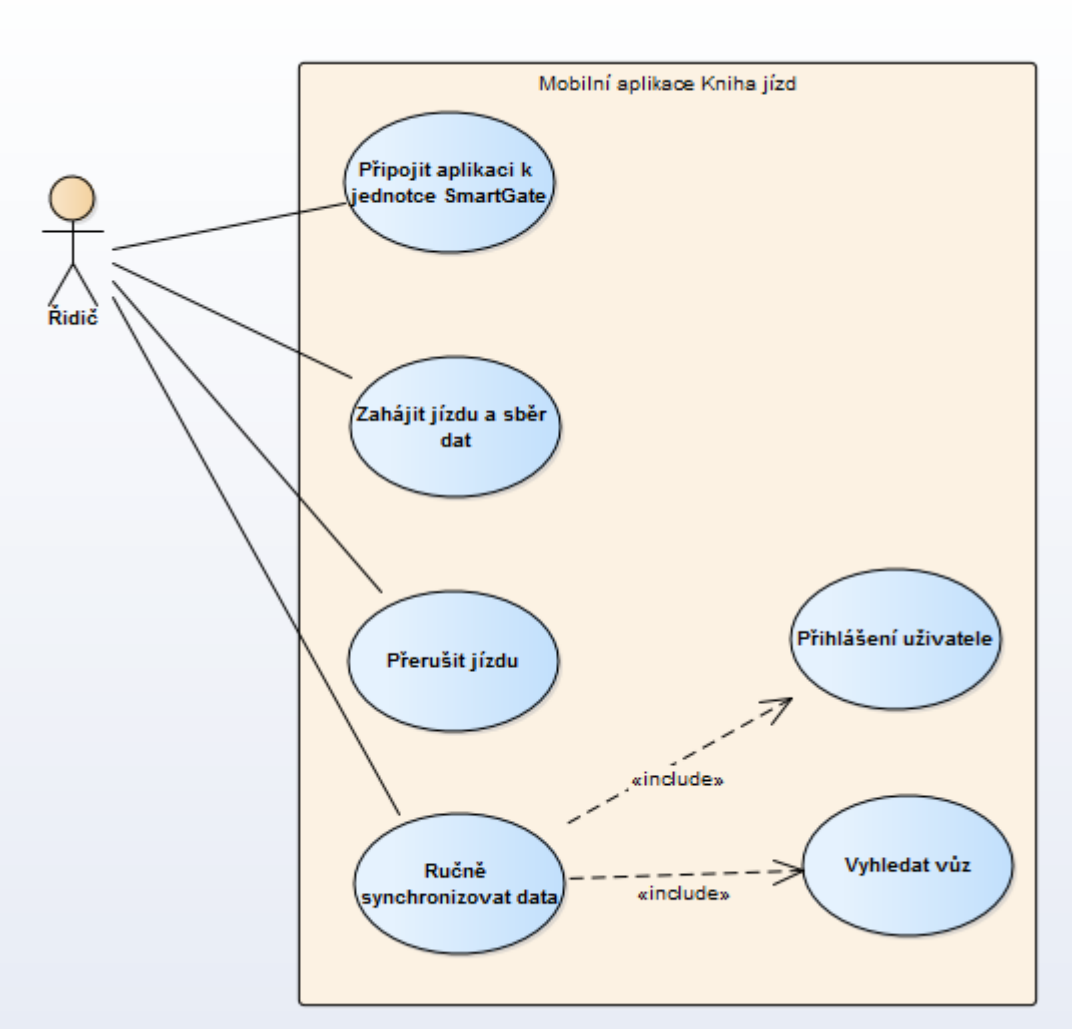
Relaci <<include>> je možné použít tam, kde ve scénáři existuje totožná část nebo sekvence, která se může opakovat ve více případech užití. Není vhodné mít stejnou část scénáře v různých případech užití. Je vhodnější tuto část vyčlenit do samostatného případu užití a vkládat jej do dalších případů užití. Tento vyčleněný případ užití je vytvořen tak, aby bylo jedno, kdo jej používá. Pomocí vazby <<include>> dodává svoje chování do základního případu užití, kde musí být definovaný přesný bod, ve kterém je řízení předáno do rozšiřujícího případu užití.

Vazba typu generalizace umožňuje společné chování pro více případů užití přemístit do rodičovských případů užití. Činí to pak model nepřehledný, proto se moc nedoporučuje využívat.

Relace <<extend>> umožňuje přidat doplňkové chování do základního případu užití. Oproti relaci <<include>> je základní případ užití i bez doplňkového chování soběstačný. Jedná se tedy o volitelné součásti.

5.4.1. Diagram případů užití pro aplikaci Kniha jízd

Následující diagram případů užití obsahuje případy užití pouze pro mobilní aplikaci, která slouží hlavně ke sběru dat a k synchronizaci dat se serverem, který je přístupný pouze přes webové rozhraní. Funkcionalita a případy užití přístupné pro řidiče přes aplikaci jsou záměrně omezeny.

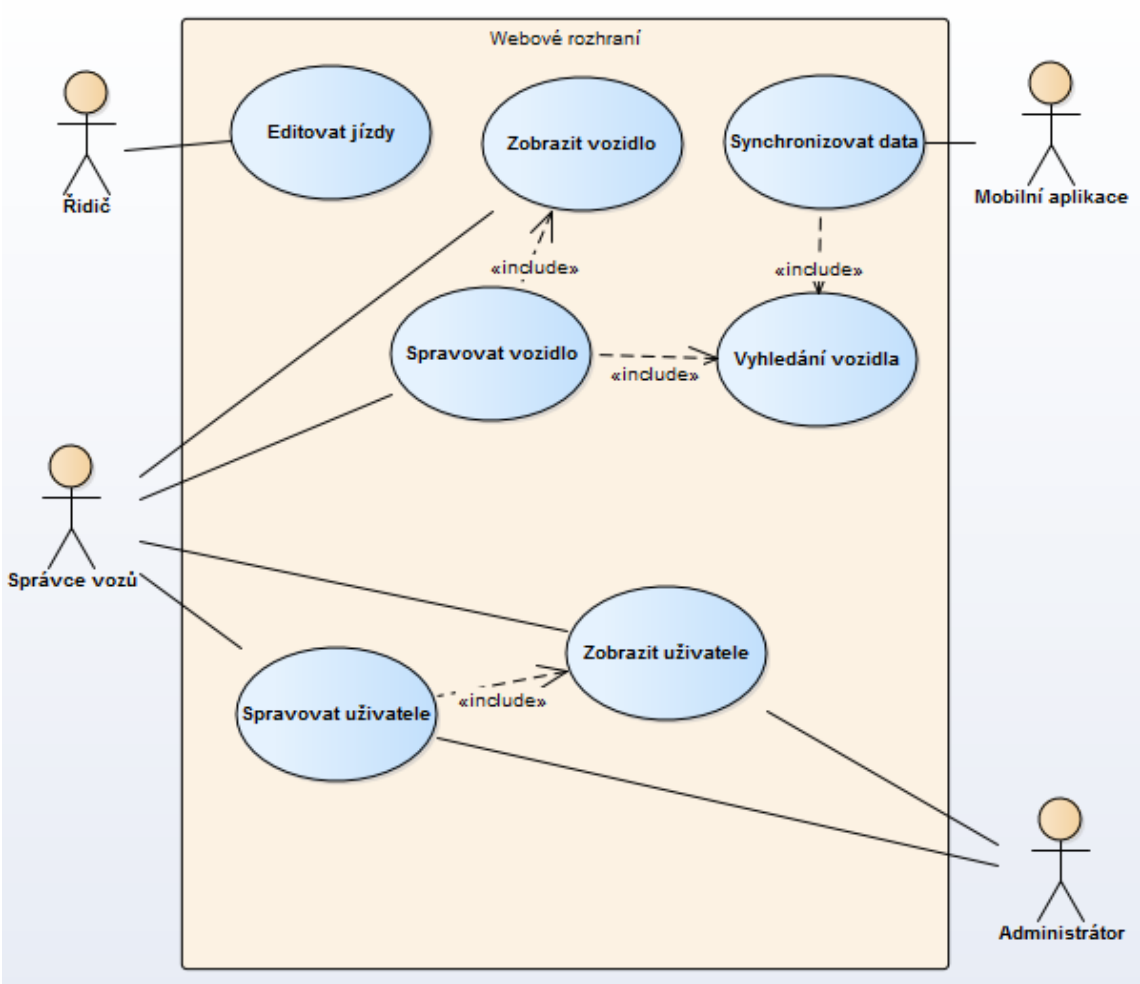


Obr. 13: Model případů užití pro aplikaci Kniha jízd

5.4.2. Diagram případů užití pro webové rozhraní aplikace Kniha jízd

Následuje diagram užití pro webové rozhraní aplikace kniha jízd. S webovým rozhraním pracují čtyři aktéři. Řidič má možnost pouze editovat svoje jízdy. Tím se myslí úprava synchronizovaných údajů, přidání popisků k jednotlivým jízdám, doplnění povinných přestávek atd. Dále se systémem pracuje pracovník v roli správce vozů, který sleduje stav vozidel, má možnost přidávat nová vozidla do systému a odstraňovat stará. Kromě toho může editovat údaje o řidičích, přidávat nové a mazat staré. Administrátor má možnost editovat správce vozů. Kromě těchto tří aktérů do webového rozhraní ještě vstupuje mobilní aplikace, která tam synchronizuje data.

Vazby <<include>> je využito všude tam, kdepřed samotnou editací vozidla, uživatelů, nebo dat je třeba tuto entitu nejprve vyhledat a zobrazit.



Obr. 14: Model případů užití pro webové rozhraní systému

5.5. Diagram tříd

Třídy a objekty jsou základním stavebním prvkem objektově orientovaných systémů. Objekt vznikne spojením dat a funkcionality za účelem plnění zodpovědností. Každý objekt má identitu, metody, vlastnosti a zodpovědnost. V průběhu analýzy se hledají objekty ve zkoumané oblasti.

Model tříd je strukturální model, jehož účelem je identifikovat základní entity, ze kterých se bude systém skládat, jejich služby, které budou nabízet okolí a jejich vztahy mezi sebou tak, aby bylo možné realizovat chování, které je dle modelu případů užití požadované. Je nutné při tom vycházet z popisu požadovaného vnějšího chování systému.

Tento model popisuje návrh struktury prvků, jejich služeb a jejich vzájemných vazeb. Pomocí navazujících modelů – modely interakce – je pak popsána spolupráce prvků struktury na realizaci požadovaného chování. K zachycení požadavků tohoto modelu se používá diagram tříd.

Diagram tříd je nejzákladnější strukturálním diagramem UML. Jeho smyslem je popsat stavební prvky systému (třídy objektů) a jejich vzájemné vztahy. Při tvorbě diagramu tříd se vychází z požadované funkcionality, která byla definována v diagramu případů užití.

Diagram tříd se skládá ze základních stavebních prvků:

- Třídy objektů, což jsou abstraktní definice množiny objektů. Třídy mají definovány následující součásti:
  - Atributy, které určují vlastnosti objektu
  - Metody, co vytvářejí chování objektu a realizují požadovanou funkcionalitu systému
- Rozhraní tříd definují kontrakt, ke kterému se třídy přihlašují
- Asociace jsou vztahy mezi objekty, na základě kterých jsou realizovány interakce vedoucí k požadovanému chování systému

Model tříd se obvykle vytváří na dvou úrovních abstrakce.

Model analytických tříd definuje základní logické celky a jejich vztahy, bez implementačních detailů. Hlavní část modelu je tvořena tzv. doménovou třídou (objekty), jejich zodpovědnost a vztahy.

Model návrhových tříd rozpracovává dále analytické třídy do podoby k implementaci. Bývají do něj doplněny rozhraní, řídící a podpůrné třídy.

Třídy se obvykle modelují na třech úrovních architektury:

- Modelování uživatelského rozhraní (Interface Class Diagram) – diagram zobrazuje strukturu uživatelského a komunikačního rozhraní, vztahy mezi jednotlivými prvky, jejich vlastnosti a metody.
- Modelování logiky aplikace (Business Class Diagram), který se používá k návrhu struktury a vztahů objektů, které so podílejí na vytváření aplikační logiky.
- Modelování podpůrných tříd, kde je diagram tříd využit k návrhu podpůrných, technologických a dalších tříd. [12]

V rámci analytického modelu se mohou objevovat hlavně analytické třídy logiky aplikace, v rámci návrhového modelu to jsou pak rozpracované třídy z aplikační logiky, třídy uživatelského rozhraní a podpůrné třídy.

Analytický model tříd má úzký vztah na realitu, je jednoduchý a soustředí se pouze na významné rysy, je srozumitelný – měl by být pochopitelný i bez dlouhého popisování, je úplný – není možné při jeho tvorbě zapomenout na žádný významný prvek.

Analytické třídy, z kterých se analytický model skládá, jsou mapované na pojem používaný v reálném systému. Jejich množina odpovědnosti je malá a jasně definovaná a většinou mají minimum vazeb na okolní třídy.

Pro vzájemnou interakci objektů je nutná jejich spolupráce. V modelu tříd existuje několik druhů relací:

- Asociace popisuje činnosti, na kterých se dvě provázané třídy podílí v podobě rolí (AUTO – řízení – OSOBA). Aby tyto třídy mohly spolupracovat, vždy musí být provázány.
    - Multiplicita popisuje četnost vzájemných výskytů objektů provázaných tříd v jednom okamžiku.

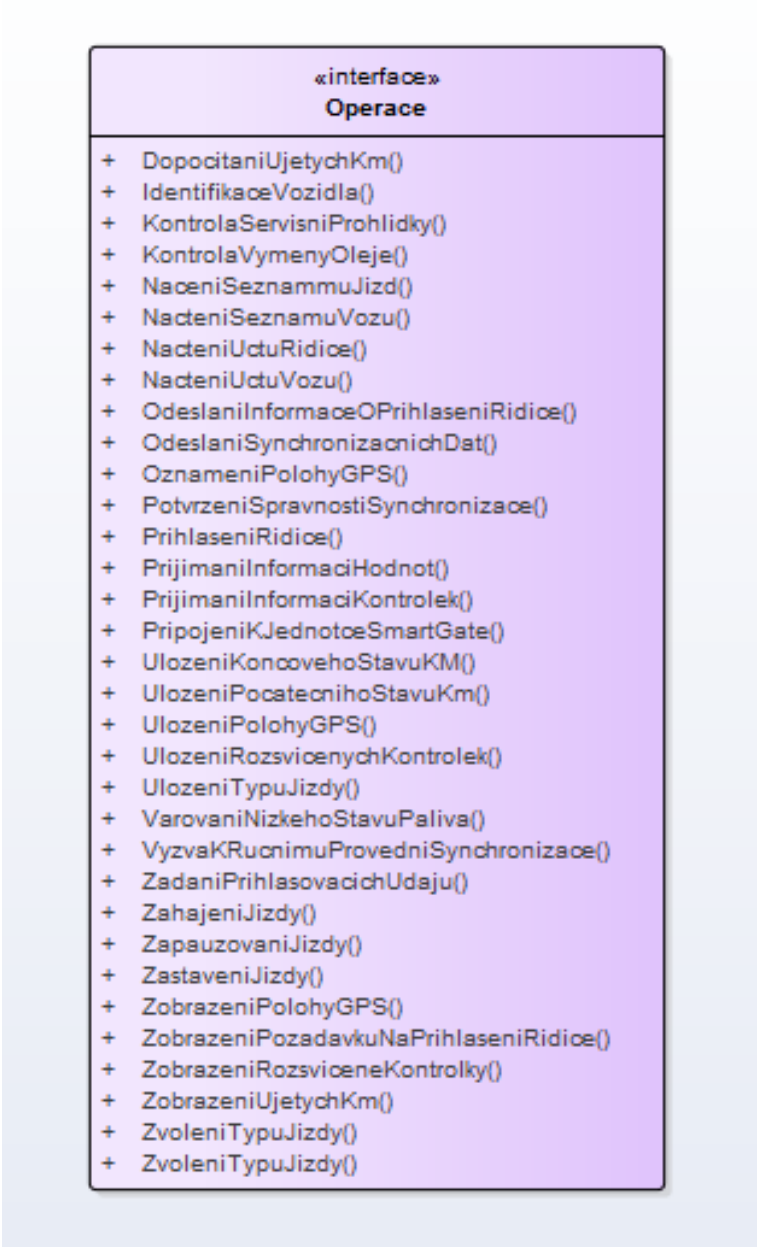
        - Kvalifikátor asociace je vlastnost, která modeluje způsob zacházení s prvky kolekce. Určuje jedinečný klíč, podle kterého je možné identifikovat prvek kolekce.
    - Generalizace – specializace vychází z podobnosti mezi třídami. Definuje vztah mezi obecnou třídou a třídou od ní odvozenou speciální třídou. Jedná se o dědičnost. Generalizace zobecňuje společné vlastnosti tříd a specializace zachycuje rozdíly.
    - Závislost je skupina relací, v nichž se změna v hlavní třídě projeví automaticky i v závislé třídě. Existuje jich více typů:
        - Use – závislá třída užívá hlavní element vlastní implementace
        - Call – metoda v závislé třídě volá metodu v hlavní třídě
        - Refine – závislá třída je další verzí hlavní třídy

Během modelování analytického modelu se nejprve identifikují základní logické celky, které by měly realizovat služby definované případy užití. Existuje na to několik metod: Metoda hledání podstatných jmen a sloves, analýza scénářů případů užití, atd. [13]

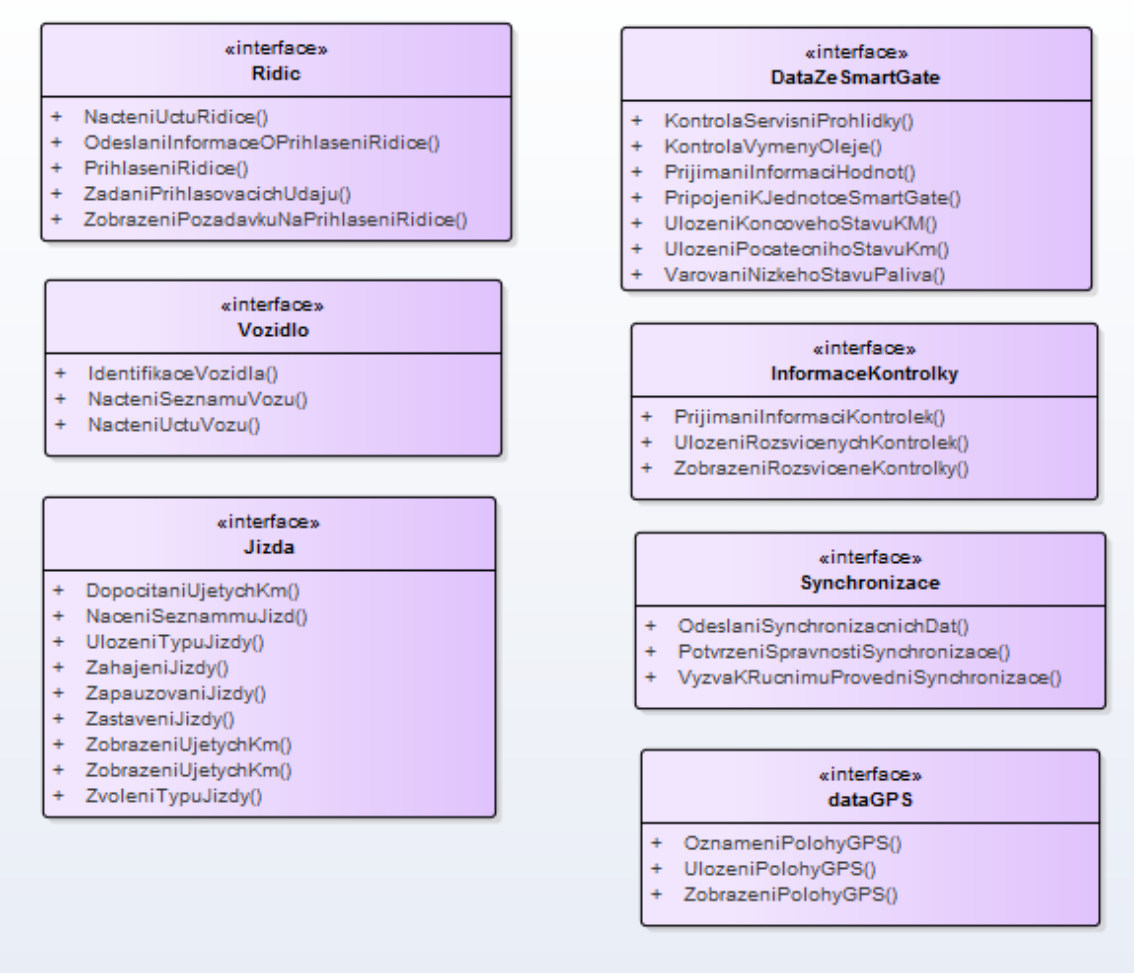
5.5.1. Diagram Tříd pro aplikaci Kniha Jízd

Ze scénářů případů užití se dají vyextrahovat názvy činností, které jsou pro běh systému nezbytné. V úvodu je třeba vypsat si metody, které jsou zodpovědné za činnosti, které v aplikaci probíhají. Tyto činnosti je třeba vložit do jedné třídy typu interface. Pro praktický příklad byly v první fázi identifikovány metody v obrázku č. 15.

V následujícím kroku je třeba rozmístit nalezené metody do analytických rozhraní, dle hlediska soudržnosti, aby metody pracovaly jen s jednou skupinou dat. Viz obrázek č. 16.



Obr. 15: Prvotní seznam metod aplikace Kniha jízd



Obr. 16: Rozdělení metod do analytických rozhraní

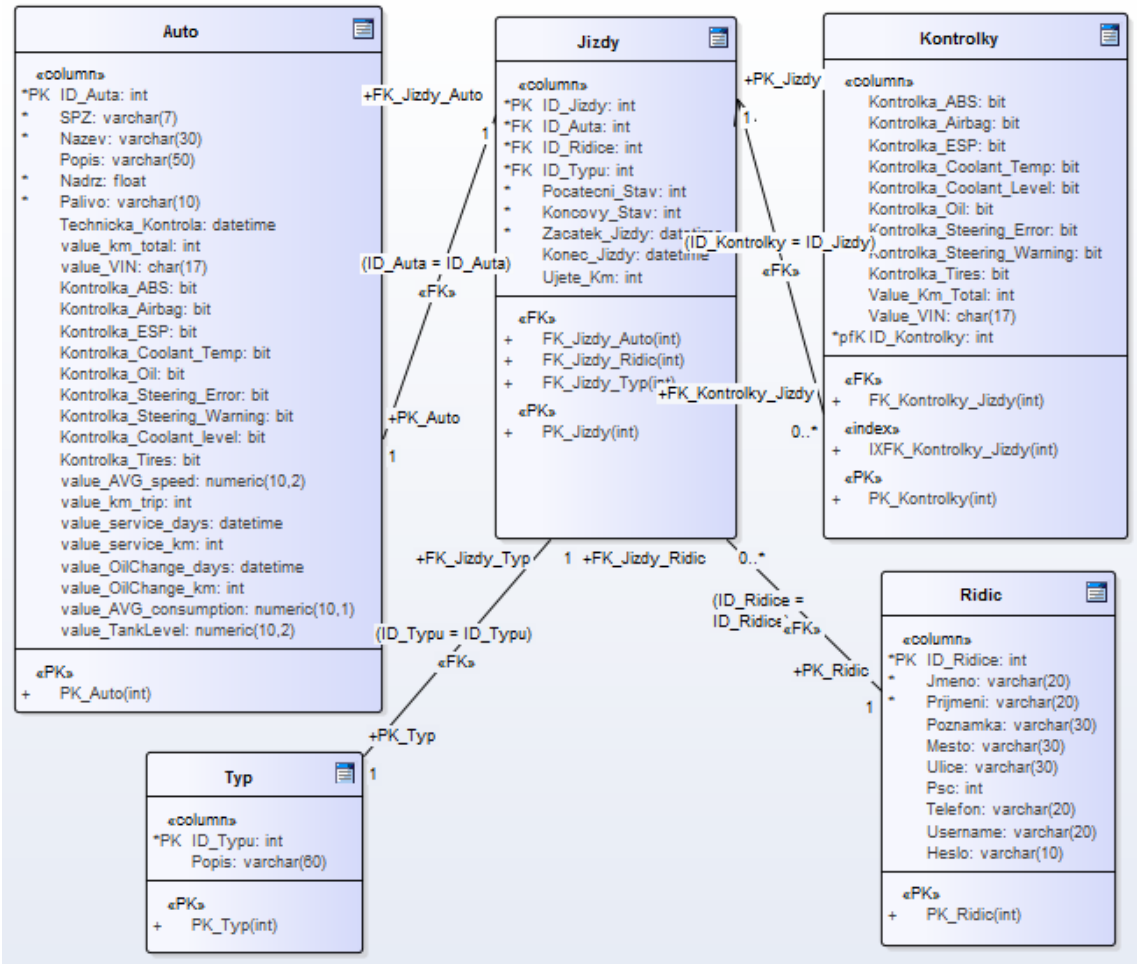
Výsledkem modelování je fyzický datový model na obrázku 17. Je to další stupeň návrhu, který umožní správně namapovat objektový model do relační databáze tak, aby objekty mohly efektivně komunikovat s tabulkami relační databáze. [13]

- Z atributů se vyvinou sloupce
- Z instance objektů řádky
- Klíče se vybírají z atributů, nebo se vytvoří extra.
- Asociace 1:M definují relaci 1:M v datovém modelu
- Asociační objektová třída způsobí vznik vazební tabulky
- Agregace mezi třídami vytvoří relaci v datovém modelu
- Kompozice vede ke vzniku relace v datovém modelu

Tento datový model je základem databáze v Microsoft SQL Server a proto se jedná o fyzický model. Je ale třeba upravit formáty sloupců, aby byly dodrženy dovolené datové typy. Dále je třeba nastavit vlastní a cizí klíče. [13]

Tyto úpravy v datovém modelu je ovšem třeba zavádět se znalostí správné implementace databáze, protože špatně navržená databáze může znehodnotit celou analýzu vyvíjeného systému. [13]

Pomocí tohoto modelu je možné automaticky vytvořit skript, který obsahuje příkazy pro MySQL, Microsoft SQL Server, nebo Oracle. Pomocí tohoto skriptu je pak vytvoření celé struktury tabulek, nastavení referenční integrity, indexů, triggerů atd. [13]



Obr. 17: Výsledný model připravený pro vytvoření databáze

5.6. Modelování firemních procesů pomocí BPMN

„Business Process Modeling Notation (BPMN) je grafická notace (soubor grafických objektů a pravidel, podle nichž mohou být mezi sebou spojovány), která slouží k modelování procesů.“ [15]

Za jejím vznikem je iniciativa BPMI (Business Process Management Initiative), jejímž hlavním cílem bylo stvořit notaci, která bude čitelná pro všechny účastníky životního cyklu procesu. Díky BPMN se úspěšně podařilo zmenšit komunikační mezeru mezi návrhem a implementací procesu a díky desítkám nástrojů, které jej používají, se stalo de facto standardem pro modelování procesů. [15]

V současné době aktuální je verze 2.0, která chce být jedinou formou notace pro tvorbu všech modelů podnikových procesů.

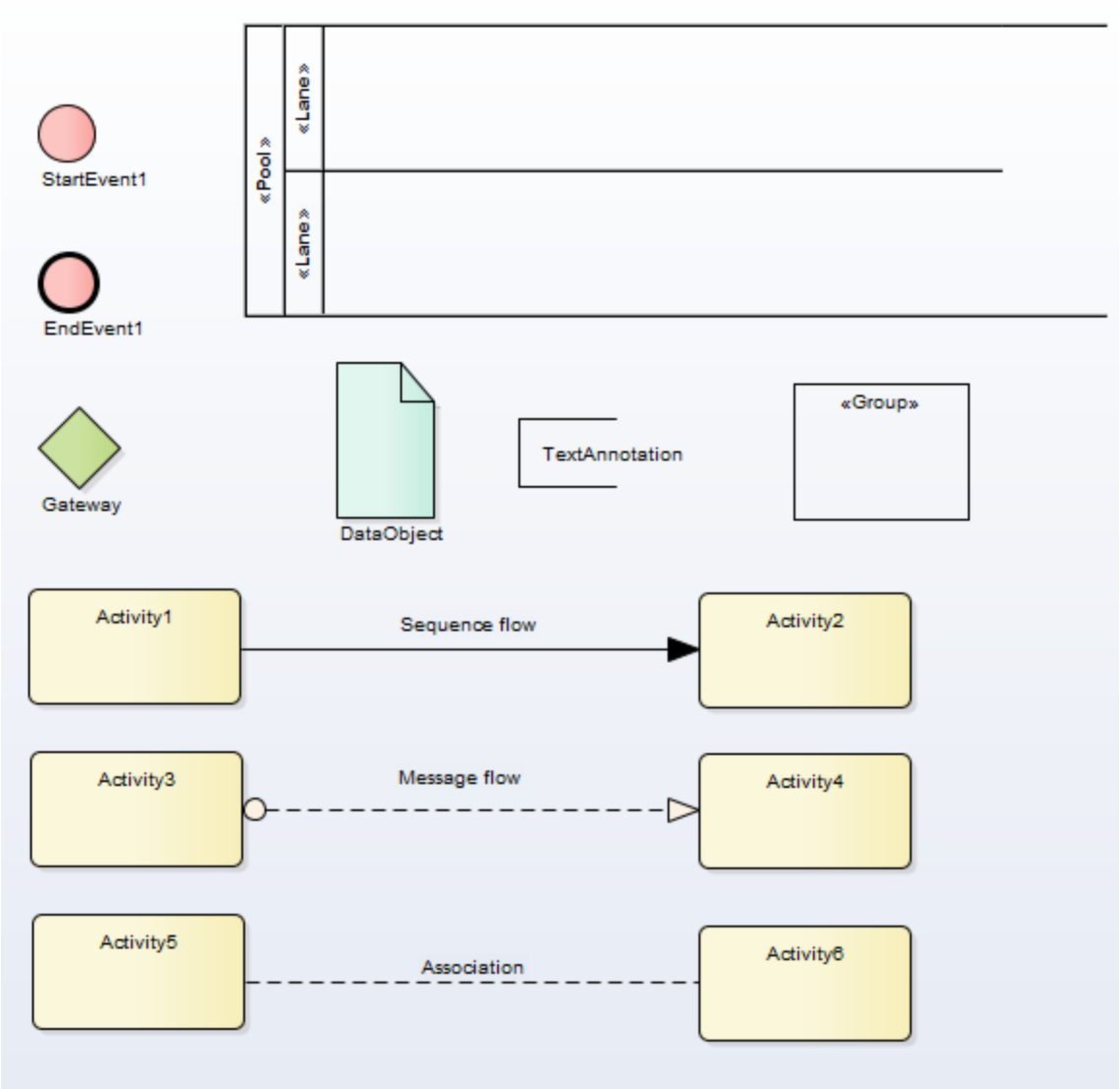
Při modelování BPMN se pracuje s několika pojmy:

- Firemní proces je množina k cíli vztažených aktivit organizace, vytváří produkt/službu, často je smyslem existence organizace, často jich je více realizováno paralelně, bez vzájemné časové návaznosti.
- Workflow je označení pro návaznost aktivit v rámci firemního procesu.
- Sub-proces je aktivita, která je dále dekomponována (vytváří hierarchie), tj. skupina dalších sub-procesů a aktivit
- Procesní vlákno je označení pro specifickou cestu přes workflow, to totiž může být tvořeno více vlákny.

Na obrázku 18 je vidět, že BPMN obsahuje 4 základní typy elementů, které se dělí ještě na další podskupiny. [15]

- Tokové objekty souvisí s tokem informací v procesu
    - Event = událost, značí se kroužkem a přímo ovlivní tok procesu. V události proces začne nebo skončí.
    - Activity = aktivita, je obdélník s kulatými rohy, znázorňuje činnost či práci. Může v sobě obsahovat další proces
    - Gateway = brána, jde o kosočtverec, který značí rozbíhání, či souběh toků procesů. Může jít o rozhodování či paralelní tok

- Spojovací objekty sloužící ke spojení tokových objektů
    - Sequence flow = sekvenční tok je nepřerušovaná čára s vyplněnou šipkou a určuje pořadí aktivit
    - Message flow = tok zpráv se značí přerušovanou čárou s prázdnou šipkou a znázorňuje tok zpráv mezi dvěma účastníky procesu
    - Association = asociace je přerušovaná čára, která spojuje objekt s nějakou dodatečnou informací
- Artefakty značí upřesňující informace pro proces, nemají vliv na jeho tok
    - Data object = datový objekt se značí ikonkou s listem papíru. Reprezentuje data, se kterými pracují aktivity
    - Group = seskupení je obdélník kreslený přerušovanou čárou, který seskupuje aktivity
    - Annotation = poznámka je text pro dodatečné informace
- Plavecké dráhy, neboli Swimlanes slouží k uspořádání činností v procesu dle rolí
    - Pool ohraničuje proces a reprezentuje účastníka procesu. Komunikace mezi pooly probíhá pomocí zasílání zpráv.
    - Lane = dráha je část poolu, která slouží ke kategorizaci aktivit. Může oddělovat například oddělení nebo funkce organizace. Komunikace mezi dráhami probíhá přes sekvenční tok

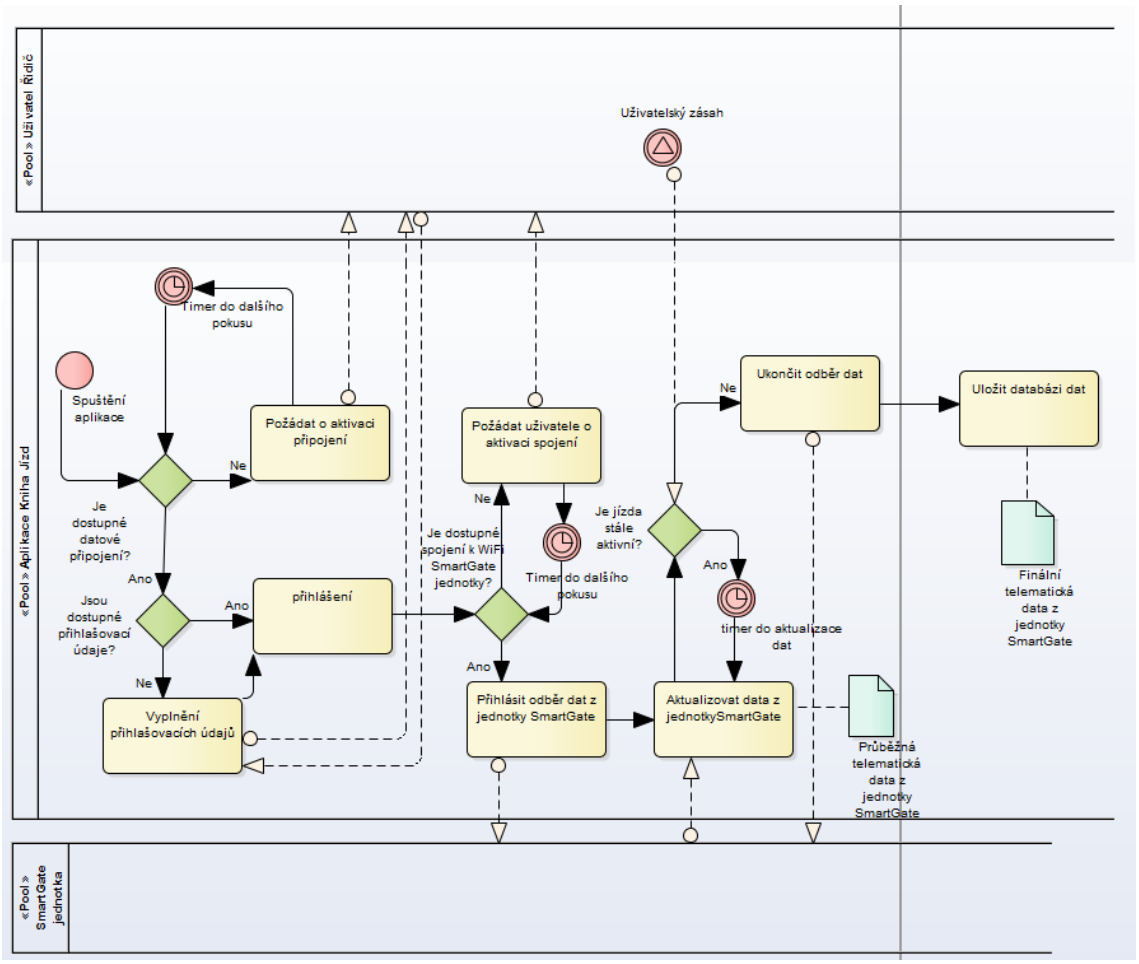


Obr. 18: Základní elementy BPMN

Tvorbu BPMN modelu je třeba začít vytvořením hierarchie procesů. Je třeba určit hlavní – kořenový proces, který představuje celé vybrané prostředí. Tento hlavní proces se rozpadne na několik základních firemních procesů organizace, které představují „core business“, tj. předmět působení organizace, tak i podpůrné procesy.

Tyto hlavní procesy se dále rozloží do hierarchie aktivit a akcí. Název každého procesu musí vždy vystihovat jeho účel a musí vyjadřovat činnost. V popisku se uvedou hlavní cíle existence procesu. Každá aktivita musí být v hierarchii pouze jednou.

Dále se identifikují jednotlivé události, které iniciují aktivity uvnitř firemních procesů. Aktivity se pak provážou pomocí Sekvenčních toků nebo dalších prvků.

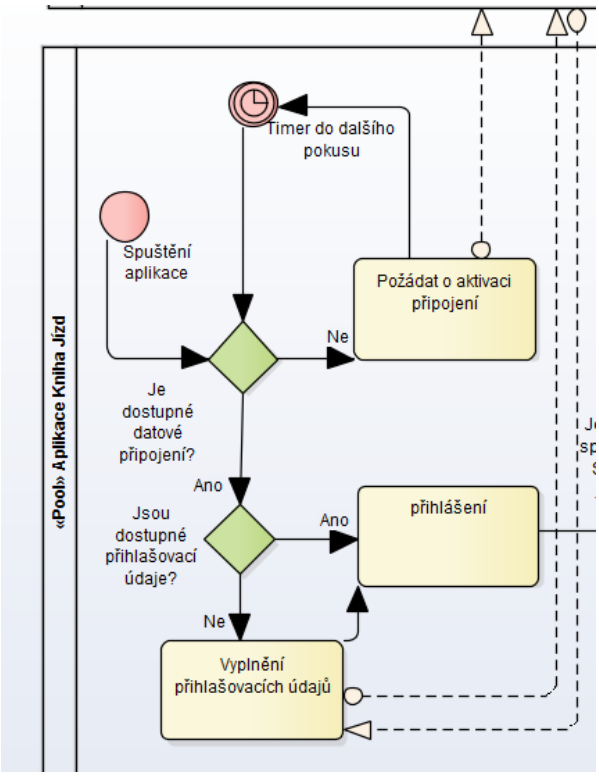


Obr. 19: Základní model BPMN spolupráce mezi uživatelem Řidič, aplikací Kniha jízd a jednotkou SmartGate

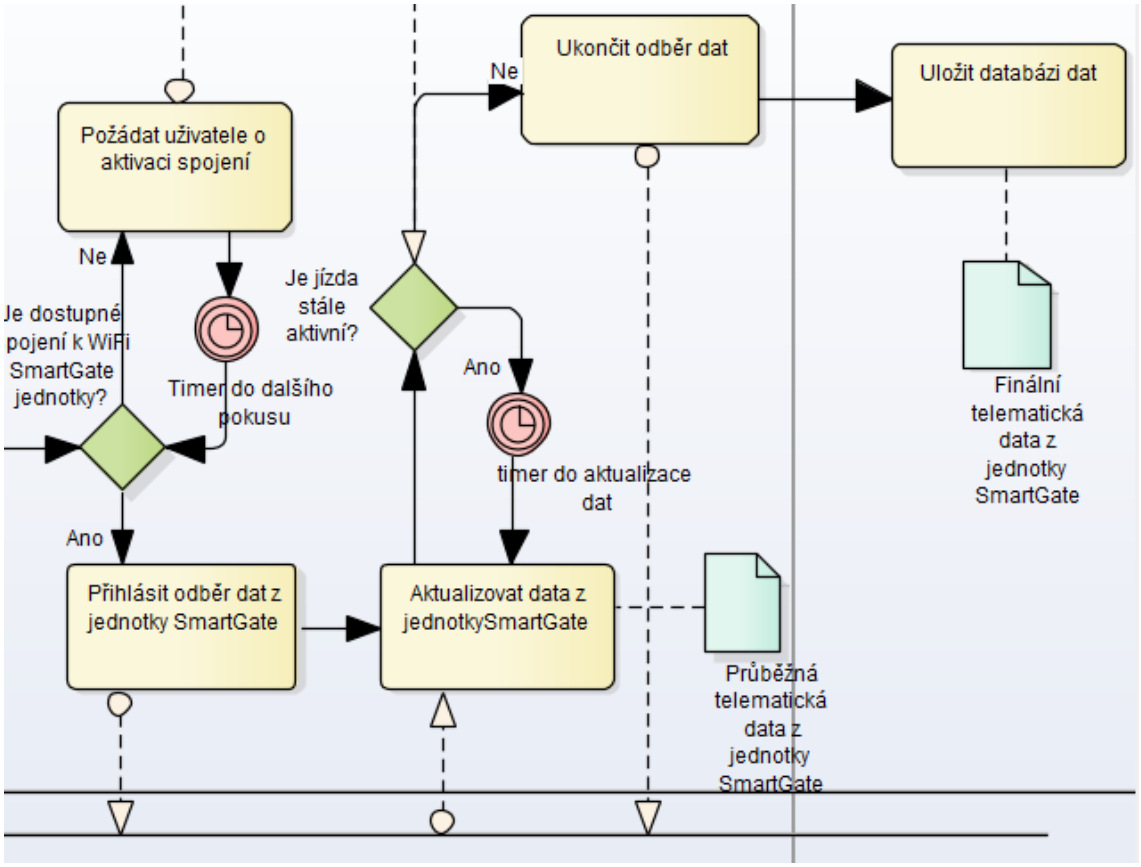
Na obrázku 19 je vidět celkový pohled na BPMN model systému aplikace elektronické knihy jízd. Obsahuje tři pooly – Jednotku SmartGate, mobilní aplikaci a uživatele řidiče.

Nejprve se řeší dostupnost mobilního připojení a přihlášení uživatele do aplikace, viz detail v obrázku č. 20. Je tam použitý interactive event timer, stejně jako jinde v modelu. Zajišťuje časový rozestup mezi pokusy o připojení. Dále se na obrázku 21 kontroluje připojení k jednotce SmartGate. Když je dostupné, tak se od ní zprávou vyjedná zasílání dat, která se pak v určitém časovém okamžiku aktualizují.

Pokud uživatel zasáhne a přeruší jízdu, vygeneruje se finální sada dat, která se může synchronizovat se serverem.



Obr. 20: Detail modelu BPMN, část přihlášení



Obr. 21: Detail modelu BPMN, část příjmu dat a ukončení

6. Závěr

V první části diplomové práce jsou uvedeny informace o vědní oblasti elektroniky osobního vozu. Je zde vysvětleno množství základních principů, které jsou nutné pro další práci s daty z vozidlových sběrnic CAN v praktické závěrečné části. Dále jsou v práci představeny aktuálně využívané nebo teprve připravované telematické systémy. V praktické části jsou vysvětleny základy jazyka UML, který je následně průběžně využíván k tvorbě modelu aplikace elektronické knihy jízd pro chytré telefony. Tato aplikace ke své práci využívá data z elektronické jednotky SmartGate, která je montovaná automobilkou Škoda Auto do většiny jejich vozů od příchodu nové (3.) generace vozu Škoda Fabia. Bohužel se nejedná o povinnou výbavu, ale pouze o příplatkovou položku. Což pro flotily vozidel, pro které je aplikace určena, není žádný problém. Navíc je jednotka s příslušenstvím v prodeji i k dodatečné montáži v servisu, takže v případě zájmu je možno tuto výbavu do vozidel přidat.

Výchozí hypotézu, že pokud existují v osobním vozidle elektronické jednotky, které poskytují informace z datových sběrnic volně mimo tyto sběrnice, pak dají se tato data využít k vylepšení již existujících řešení telematických systémů, se podařilo potvrdit vytvořením modelu aplikace, která tato data využívá k vylepšení telematického systému elektronické knihy jízd.

Pro připravení modelu aplikace a synchronizačního serveru tak, aby se podle něj dalo systém správně objektově naprogramovat, bude vyžadovat ještě další práci na dopracování stávajících modelů a připravení dalších, které jsou pro správný vývoj nezbytné. V tuto chvíli není možné tuto aplikaci realizovat, protože chybí API ke komunikaci s jednotkou SmartGate, které nebylo od firmy Škoda Auto zatím uvolněno.

Seznam použitých zdrojů

[1] Sběrnice CAN. *Elektrorevue* [online]. Ústav telekomunikací: VUT FEKT Brno, 2003, 16. 6. 2003 [cit. 2015-12-26]. Dostupné z: http://www.elektrorevue.cz/clanky/03021/index.html

[2] *CAN specification*. Version 2.0. Postfach 50, D-7000 Stuttgart 1: Robert Bosch GmbH, 1991, 72 s.

[3] Introduction to CAN. *Vector Academy: Working knowledge* [online]. Ingersheimer Straße 24, 70499 Stuttgart, Germany: Vector Informatik GmbH, 2010-01-04 [cit. 2015-12-26]. Dostupné z: http://elearning.vector.com/vl_can_introduction_en.html

[4] VLK, František. *Automobilová elektronika 2: Systémy řízení podvozku a komfortní systémy*. 1. vyd. Brno: František Vlk, 2006, 308 s. ISBN 80-239-7062-3.

[5] *FLEET firemní automobily: Produktová příloha - telematika ve službách vozových parků*. Česká republika: Club 91 s.r.o., 2015, **2015**(2). ISSN 1214-861X.

[6] Konektivita: Smartphone na čtyřech kolech. *Škoda Auto* [online]. Mladá Boleslav, 2015 [cit. 2015-12-30]. Dostupné z: http://www.skoda-auto.cz/models/nova-fabia/konektivita/

[7] Google Play: ŠKODA LittleDriver. *Google Play store* [online]. ŠKODA AUTO a.s., 14. září 2015 [cit. 2015-12-30]. Dostupné z: https://play.google.com/store/apps/details?id=cz.skodaauto.littledriver

[8] ECall (automatické tísňové volání z vozidla). *Český kosmický portál: Informační stránky Koordinační rady ministra dopravy pro kosmické aktivity* [online]. Praha, 2015 [cit. 2015-12-29]. Dostupné z: http://www.czechspaceportal.cz/3-sekce/its---dopravni-telematika/ecall/

[9] *Nařízení Evropského parlamentu a Rady (EU) 2015/758 ze dne 29. dubna 2015: o požadavcích na schválení typu pro zavedení palubního systému eCall využívajícího linku tísňového volání 112 a o změně směrnice 2007/46/ES*. In: . Strasbourg: Evropský Parlament a Rada EU, 2015, číslo (EU) 2015/758; Celex 32015R0758.

[10] HeERO. *Technické řešení pilotního projektu eCall*. Praha, 2012. Dostupné také z: http://www.czechspaceportal.cz/files/files/eCall%20sekce/promotion/letak_eCall_CZ_print.pdf

[11] Smart Racer. *Google Play* [online]. 14. října 2014 [cit. 2015-12-31]. Dostupné z: https://play.google.com/store/apps/details?id=com.fuerteint.smartracer

[12] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007, 567 s. ISBN 978-80-251-1503-9.

[13] KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. Vyd. 1. Brno: Computer Press, 2004, 158 s. ISBN 80-251-0231-9.

[14] *Unified Modeling Language: Infrastructure*. Version 2.0. OMG - Object modelling group, 2006. Dostupné také z: http://doc.omg.org/formal/2005-07-05.pdf

[15] *BPM prakticky: 3. část: Úvod do BPMN* [online]. Praha: BPM prakticky, 2008 [cit. 2016-01-05]. Dostupné z: http://bpm-sme.blogspot.cz/

Seznam použitých zkratek

CAN	Controller Area Network
ISO	Mezinárodní organizace pro normalizaci
ABS	Protiblokovací systém
GPS	Global Positioning System
CASE	Computer-aided software engineering
UML	Unified Modelling Language
SEP	Software Development Process
Wi-Fi	Wireless fidelity – bezdátová síť standardu IEEE 802.11

Seznam tabulek, obrázků atd.

Seznam obrázků:

Seznam tabulek:

Evidence výpůjček

Prohlášení:

Dávám svolení k půjčování této diplomové práce. Uživatel potvrzuje svým podpisem, že bude tuto práci řádně citovat v seznamu použité literatury.

V Praze,

Jan Vulc

Jméno	Katedra / Pracoviště	Datum	Podpis