Mendelova univerzita v Brně
Provozně ekonomická fakulta

Adaptivní studijní asistent na bázi lokačních služeb

Diplomová práce

Vedoucí práce:
Ing. David Procházka, Ph.D.

Bc. Ivo Pisařovic

Brno 2018

Zde bude vloženo zadání práce

Poděkování

Chtěl bych na tomto místě poděkovat vedoucímu mé práce Ing. Davidu Procházkovi, Ph.D., za cenné rady, připomínky a podporu. Velký dík také patří celé mé rodině za podporu během celého studia.

Čestné prohlášení

Prohlašuji, že jsem práci: **Adaptivní studijní asistent na bázi lokačních služeb** vypracoval samostatně a veškeré použité prameny a informace uvádím v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*

Jsem si vědom, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 15. 5. 2018 .....

Abstract

Pisařovic, I. Adaptive study assistant based on location services. Diploma thesis. Brno, 2018.

This thesis describes design and implementation of a system with natural user interface that helps students to fulfil their study duties. It improves their life in the university campus by helping them with common problems. The system identifies the user and the user's current location and behaviour. On the basis of this information, the system provides content relevant to the particular user. The content is displayed on the user's mobile device or on a corridor display within the vicinity of the user. The user can interact with the content, e.g. request further information about the topic or navigate to the mentioned event.

Keywords

university, assistant, location based service, mobile device, user interface, display

Abstrakt

Pisařovic, I. Adaptivní studijní asistent na bázi lokačních služeb. Diplomová práce. Brno, 2018.

Práce se zabývá návrhem a tvorbou systému s přirozeným uživatelským rozhraním, který pomáhá studentům plnit jejich povinnosti včas, správným způsobem a zpříjemňuje jim život v prostředí univerzity řešením častých problémů. Systém identifikuje uživatele, jeho polohu a chování. Na základě těchto údajů vybírá relevantní obsah podle potřeb daného uživatele. Relevantní obsah je zobrazen na mobilním zařízení uživatele nebo na displeji na chodbě poblíž uživatele. S obsahem lze dále interagovat, např. zobrazit si další informace nebo se přepnout do navigace.

Klíčová slova

univerzita, asistent, lokační služba, mobilní zařízení, uživatelské rozhraní, displej

Obsah

1 Úvod

Studenti tráví mnoho času v univerzitním kampusu. Při tom plní řadu studijních povinností a řeší organizační problémy. Noví studenti mají například problém se orientovat v množství budov a učeben, nerozumí pořádně systému hodnocení apod. Stávající studenti jsou v těchto záležitostech již zběhlí, nicméně řeší jiné problémy, jako třeba včasné a správné odevzdání závěrečné práce nebo vyhnutí se každodennímu čekání ve frontě na oběd. Pokud nedokáží studenti vyřešit problém sami nebo za pomocí svých spolužáků, obrací se většinou na webové stránky školy a také na studijní oddělení, které je zahlceno dotazy.

Na univerzitě mám možnost pracovat na projektu *Smart PEF*. Jedná se o převzetí nápadů a technologií *internetu věcí (IoT)* a *chytrých měst (Smart Cities)* do prostředí univerzity. Na základě brainstormingů se studenty byly shromážděny a vybrány jejich nejčastější problémy. Je kladen důraz na to, aby byly řešeny opravdu praktické problémy a aby se nejednalo jen o bezvýznamné ukázky technologií, jak tomu často bývá v případě *IoT* a *Smart Cities*. Věříme, že se nám podaří zlepšit život studentů v univerzitním kampusu a zahájíme novou dimenzi aplikace chytrých technologií – chytrých univerzit (*Smart Universities*).

Moje diplomová práce úzce souvisí se *Smart PEF*. Budu se zabývat návrhem asistenta, který bude studentům pomáhat s administrativními povinnostmi, studijní agendou a běžnými činnostmi v univerzitním kampusu. Aby mohl být student vhodně informován, je potřeba mu informace podávat ve správnou chvíli, na správném místě a správným způsobem. Budu se proto zabývat různými způsoby předání informace uživateli pomocí různých zařízení patřících uživateli nebo přítomných v areálu univerzity. Obsah musí být vhodně přizpůsoben potřebám uživatele, aktuálnímu času, lokaci uživatele a také potřebám ostatních uživatelů v nejbližším okolí. Uživatel bude mít možnost s obsahem také interagovat.

1.1 Cíl práce

Cílem práce je navrhnout a implementovat systém, který bude pomáhat studentům s jejich administrativními povinnostmi a problémy běžnými při životě v univerzitním kampusu. Hlavní funkcí systému je cílení obsahu správným uživatelům na správném místě. Cílení bude probíhat na základě informací o uživatelích, jejich lokace, času a parametrů prostředí. Obsah se bude zobrazovat na displejích na chodbách a v mobilních zařízeních. Uživatel bude mít možnost s obsahem interagovat.

K dosažení cíle práce bude vytvořen přehled existujících projektů využívajících lokační služby. Důraz bude kladen zejména na způsob využití lokačních služeb a způsob předávání obsahu uživateli. Dále bude vytvořen přehled služeb, technologií a zařízení, které mohou být využity pro implementaci systému. Součástí rešerše bude také přehled principů tvorby uživatelského rozhraní. Na základě těchto přehledů budou navrženy metody předávání relevantního obsahu studentům na základě jejich potřeb, lokace a času.

Navržené metody budou ověřeny implementací systému, který bude řešit vybrané problémy související se studiem. Systém bude zahrnovat aplikace pro různé typy zařízení (mobil, tablet, velké displeje na chodbách, malé e-inkové displeje u učeben). Je nutné zajistit obousměrnou komunikaci mezi mobilními zařízeními a displeji na chodbách pro možnost interakce uživatele s obsahem a pro získávání informací o uživatelích v okolí displeje. Testovací mobilní aplikace bude cílena na platformu *iOS* vzhledem k mým předchozím zkušenostem. Navržené řešení však bude univerzální a snadno implementovatelné na jakékoliv platformě.

V závěru práce bude vyhodnocena přívětivost a použitelnost uživatelského rozhraní, jak z pohledu autora, tak na základě testování na uživatelích, a také bude diskutována využitelnost systému v praxi.

2 Rešerše

V první části rešerše jsou popsány existující projekty a koncepty, které využívají lokační služby k interaktivní akci s uživatelem a dále jsou identifikovány použité metody. Další části rešerše se věnují teoretickým konceptům uživatelských rozhraní a výzkumem zabývajícím se chováním uživatele v interakci s digitálním obsahem. Zbývající části se věnují technologiím potřebným pro implementaci lokační aplikace a komunikaci mezi zařízeními.

2.1 Existující projekty

Využití lokačních služeb

Z různých zdrojů jsem shromáždil asi 200 případů užití lokačních služeb pro interakci s uživatelem. Z případů jsem vybral pouze ty, které jsou přenositelné do prostředí univerzity, ať už z hlediska obsahu nebo z hlediska technického řešení. Podobné případy užití jsem sloučil.

1. Nabízení slev poblíž zboží – např. obchodní řetězec *Tesco Lotus*, prodejny *Apple Store* nebo letiště *Hamad International Airport*; marketing a sběr dat o zákaznících na úrovni celých měst – např. město Cleveland. (Tesco Lotus, 2018; Křížová, 2017; Hamad International Airport launches mobile app with iBeacon features, 2016)
2. Do obchodu přichází zákazník, prodavačům se zobrazí informace o zákazníkovi ještě než vstoupí. (Alexandru, 2015)
3. Označení nějakého místa, aby ho pak našel kamarád. (Alexandru, 2015)
4. Lokační soutěžení: focení míst, úkoly, získávání odznaků. Díky aktivitě uživatelů se sbírají cenná data o okolí a lidech – např. geolokační hra *Zlatý Písek* (Křížová, 2017)
5. Detekce čekání ve frontě a optimalizace – např. čerpací stanice v Abu Dhabi nebo stadion *Camp Nou FC Barcelona*. (Duque, 2015; Al-Rajab, Alkheder a Hoshang, 2017)
6. Rozvrhy hodin při přiblížení se k učebně. (Alexandru, 2015)
7. Sledování docházky – např. aplikace *BeHere*. (Mallik, 2015)
8. Navigace do správného terminálu na letišti, navigace v metru nebo na nádraží – např. železnice *MTR Crossrail of London*, letiště *Hamad International Airport* nebo aplikace *At the Ballpark*. (Mittal, 2017; Hamad International Airport launches mobile app with iBeacon features, 2016; Mallik, 2015)
9. Notifikace uživatelů, kteří v nesprávný čas budou příliš daleko od vstupní kontroly a nemuseli by to stihnout. (Alexandru, 2015)

10. Notifikace řidičů o utíkajících cestujících. (Alexandru, 2015)
11. Sledování pozice lidí ve velkých budovách. (Duque, 2015)
12. Navigační pomoc pro handicapované – např. pomoc těhotným ženám v metru v Jižní Korei. (Mittal, 2017)
13. Doporučení jídla podle preferencí zákazníka, doporučení v čase, kdy uživatel běžně obědvá nebo večeří – např. obchodní řetězec *58 Marc's stores in Ohio* (Mittal, 2017)
14. Informace v muzeu jsou doručovány do mobilu na základě času, délky návštěvy, lokace – např. *Guggenheim Museum* v New Yorku nebo zoologická zahrada v Los Angeles. (Anderson, 2017; Křížová, 2017)
15. Doručování zpráv uživatelům pouze do vybraných částí stadionu – např. stadion *Camp Nou FC Barcelona* (Duque, 2015)
16. Spuštění konkrétní aplikace na určitém místě – např. aplikace *Launch Here*. (Mallik, 2015)
17. Správce úkolů automatizovaný pomocí lokačních služeb – např. aplikace *Proximitask*. (Mallik, 2015)
18. Poznávání lidí v okolí – aplikace *Mingleton* nebo *Google Nearby*. (Mallik, 2015; Nearby, 2018)
19. Zvýšení povědomí návštěvníků o historicky důležitých místech – např. historické území *Khalifatabad* v Bangladéši. (Rahaman, Biswas, Nazimuddin, Rahman a Khan, 2016).

Na základě výše uvedených řešení bude v dalších kapitolách práce navrhnuta aplikace lokačních služeb v prostředí univerzity. Pro lepší přehlednost jsem podobné případy užití dále zobecnil do následujících skupin:

- *Zobrazení informace* nebo doporučení uživateli, jakmile se dostane na místo, přejde po nějaké trase, nebo v určitý čas.
- *Vykonání akce* související s uživatelem, jakmile se dostane na místo (např. označení úkolu jako splněný, změna nastavení telefonu apod.).
- *Lokalizace a navigace* uživatele uvnitř velkých budov – na místo nebo k jinému uživateli.
- *Analyzování* a optimalizace toku uživatelů a front nebo detekce přítomnosti konkrétního uživatele.
- *Soutěže* založené na lokaci.

Metody adaptace obsahu

Na základě průzkumu aplikací z Google Play, iTunes a dalších zdrojů (Google Play, 2017; iTunes, 2017; Choi, Song, Lee a Bae, 2013; Broussard, 2016) byly identifikovány následující způsoby změny obsahu a předání obsahu uživateli po změně lokace.

- Notifikace s odkazem na web nebo do mobilní aplikace (lokační a časové upomínky v aplikaci *Google Keep*, slevové nabídky obchodního řetězce *Carrefour* a mnohé další).
- V aplikaci se předvyplní formulářová políčka aktuálním umístěním, např. ve vyhledávači (např. mobilní aplikace s jízdními řády *IDOS*).
- V aplikaci se na základě lokace zobrazí widget s dodatečným obsahem (vyhledání „počasí“ ve vyhledávači *Google*).
- Obsah v aplikaci je tvořen pouze bloky vybranými podle aktuální lokace a kontextu. (Chytří asistenti typu *Google Now*, *Apple Siri* nebo chatbot vybírají obsah relevantní k aktuální pozici uživatele.)
- V mobilní aplikaci je možné přepínat mezi výčtem položek a mapou. Výčet bývá řazený podle vzdálenosti od uživatele, stejně tak na mapě může uživatel přejít na svou polohu (např. aplikace pro zahraniční studenty *Erasmus in Czech Republic*).
- Jednotlivé aplikace nebo moduly jsou spouštěny automaticky nebo doporučovány podle lokace (*Launch Here*).
- Doma vyžaduje uživatel jiné informace než v práci. Obsah mobilní aplikace může být velmi efektivně rozlišen na pracovní a soukromý. Podobně se může měnit i nastavení zařízení jako je tapeta nebo třeba zvukový profil (např. aplikace *Llama - Location Profiles*).

2.2 Uživatelské rozhraní

Při návrhu aplikace bude nutné se držet základních principů tvorby uživatelských rozhraní (Nielsen, 1995; Tidwell, 2011; Hoober a Berkman, 2012; Cooper, 2014; Mortensen, 2017), které jsou obecně platné a aplikovatelné i v případě mé aplikace bez standardního uživatelského rozhraní. Vybral jsem několik zásad, které se dají aplikovat na mnou navrhovaný systém.

Abstrakce od implementačních detailů

Při návrhu systému je nutné se abstrahovat od implementačních detailů, které by nás příliš svazovaly s konkrétním technickým řešením. To by vedlo k tomu, že navržené řešení by bylo snadné na implementaci, ale méně naplňující skutečné potřeby uživatele, které by nebyly správně identifikovány. (Tidwell, 2011)

Viditelnost aktuálního stavu

Uživatel by vždy měl mít pocit, že ví, kde se zrovna nachází v rámci systému. Systém by měl poskytovat vodítka naznačující kontext sdělených informací a možnosti další interakce. (Nielsen, 1995)

Shoda s reálným světem

Systém by se měl snažit napodobovat reálný svět vizuálně i způsoby interakce (Nielsen, 1995). Hoober a Berkman (2012) uvádí jako příklad zobrazení detailních informací pomocí 3D efektu otočení bloku s obsahem na druhou stranu. Místo klasického kliknutí a přechodu na detail jsou informace zobrazeny mnohem přirozenější cestou bez kognitivně náročného přechodu na jinou obrazovku.

Kontrolovatelnost uživatelem a pocit jistoty

Uživatelé často omylem nebo záměrně ovládají systém nesprávným způsobem. Systém by měl navozovat pocit jistoty a uživatel by se neměl v systému ztratit. To může být docíleno umístěním jednoznačného prvku pro návrat na známou domovskou obrazovku. (Nielsen, 1995; Tidwell, 2011)

Konsistence a zvyky

Uživatelé očekávají od systému jazyk a prvky, které již znají z jiných podobných systémů. Důležité je proto používat běžnou a jednotnou terminologii pro klíčové části systému. Uživatelé ocení, pokud na ně bude systém mluvit jejich jazykem, jazykem specifickým pro danou doménu. (Nielsen, 1995; Tidwell, 2011; Mortensen, 2017)

Předcházení chyb

Lepší než chybová hláška je vůbec se do chybové situace nedostat. Autoři doporučují předcházet chyby např. pomocí předvyplnění políček ve formuláři, průběžné kontrole vyplňování, možnosti vrátit předchozí akci apod. (Nielsen, 1995; Tidwell, 2011)

Postupné zdokonalování schopností uživatelů

Systém by měl umožnit využití dovedností uživatelů, které jsou doménově specifické. Měl by počítat s tím, že systém bude využíván začátečníky v rámci dané domény i experty. Začátečníkům by měl naznačit jasnou cestu skrz systém a tím usnadnit jejich první setkání se systémem. Pro experty pak může systém nabídnout pokročilejší funkce, více možností naráz, složitější ovládací rozhraní a možnost přizpůsobení. Příklady uvažování uživatelů jsou na obrázku 1. (Cooper, 2014)

Obrázek 1: Příklady uvažování uživatelů při různých úrovních znalosti systému. Přeloženo do češtiny a mírně upraveno do kontextu univerzity. (Cooper, 2014)

Vodítka

Vodítka a akcelerátory mohou pomoci uživatelům s orientací v aplikaci nebo slouží pro objevení rozšiřujících funkcí. Uživatel, který je v nějakém okamžiku ztracen a neví, jak pokračovat nebo jak provést nějakou akci, začne prozkoumávat méně důležité části rozhraní, kde je tedy vhodné místo pro umístění takových vodítek. Vodítka jsou často podprahová a stačí správné vizuální odlišení tlačítka pro naznačení správné akce. (Nielsen, 1995; Tidwell, 2011)

Přímá interakce

Mortenson (2017) popisuje tři klíčové prvky interakce. *Přímost* (*directness*) naznačuje okamžitou reakci uživatelského rozhraní na akci uživatele. *Vysokofrekvenční interakce* zajišťuje reakce systému bez ohledu na stav aplikace (k posunutí obrazovky dojde při skrolování i tehdy, když není kam skrolovat, uživatel je na konci dokumentu). *Kontextová interakce* zajišťuje zjednodušení interakce za pomocí informací o kontextu a automatizace. (Mortensen, 2017)

Nízká kognitivní zátěž

Uživatel by neměl být nucen příliš přemýšlet o uživatelském rozhraní. Přemýšlení uživatele zvyšuje jeho kognitivní zátěž a nepohodlí práce se systémem. Při tvorbě rozhraní bychom se měli zaměřit na co nejrychlejší splnění potřeby uživatele. (Mortensen, 2017) K tomu můžeme využít vzory jako je *Instant Gratification*, *Good Defaults*, *Deferred Choices* nebo *Incremental Contruction* (Tidwell, 2011).

Interakce s kontextem (okolím)

Pro mobilní zařízení je kontext systému velmi důležitý pro vytvoření kvalitního uživatelského rozhraní. Znalost informací z okolí uživatele (lokace, teplota, další uživatelé) je velmi klíčová pro zjednodušení rozhraní a snížení kognitivní zátěže. U mobilních zařízení i displejů na chodbách se uživatel nezdržuje delší dobu, často je využívá za chůze a je tedy třeba používání aplikace co nejvíce zjednodušit a nevyžadovat od uživatele zadávání informací, které lze získat automaticky nebo požadovat od uživatele výběr, který může být rozhodnut automaticky. (Mortensen, 2017)

Vzhledem k tomu, že navrhovaný systém je velmi nestandardní, je potřeba důsledně dodržovat zásady z výše zmíněné literatury, a to zejména ty, které souvisí s přirozenými uživatelskými rozhraními. Jen při správném pochopení chování uživatele bude možné navrhnout tento speciální systém tak, aby byl pro uživatele prospěšný a aby uživatelé se systémem interagovali. Při špatném návrhu rozhraní by mohlo dojít k naprosté ignoraci systému nebo sledování obsahu bez interakce a tím pádem i bez důležité zpětné vazby.

2.3 Chování uživatelů a okolí systému

Adaptace obsahu a procesů v mobilních zařízeních na základě okolí se obecně označuje jako *context awareness* (2017). Při užším vymezení se pak jedná o *location awareness* (2017), tedy vlastnost zařízení umožňující se lokalizovat a tuto informaci využít pro adaptaci obsahu v aplikacích. Jednou z typických aplikací *context awareness* je např. práce Liu, Wu a Wang (2014). Autoři popisují doporučování aktivit uživatelům na základě trasy jejich pohybu a míst zájmu.

Vzhledem k tomu, že má aplikace je cílena do kampusu univerzity, kde se běžně pohybuje více lidí, je důležité kromě samotného uživatele a jeho potřeb vzít v potaz také ostatní uživatele, čas, lokaci a obecně kontext (Adomavicius a Tuzhilin, 2014). Autoři se v knize zabývají velmi podrobným teoretickým popisem systémů pro doporučování a způsoby modelování okolí systému (a obecně *kontextu*). Parametry okolí by měly být pouze jako doplňující faktor při rozhodování o obsahu, protože okolí je velmi komplexní a nelze být tak snadno identifikováno jako samotný uživatel.

Vossen a kol. (2017) popisují na obecné rovině klasifikační framework využívající beacony. Autoři popisují různé metody využití beaconů v reálných aplikacích při využití různých informací o uživateli a jeho chování. V práci jsou také popsány dva způsoby výběru obsahu na základě informací o uživateli – výběr obsahu pomocí definovaných pravidel *event-condition-action* nebo s využitím párování v ohodnoceném bipartitním grafu. Jedná se o kvalitní metodický přehled, autoři však neberou v potaz omezení, která se vyskytují při implementaci.

V článku od autorů Feng a Liu (2015) je namodelováno uživatelské rozhraní a následně je dokázáno, jak ho lze s využitím lokačních služeb udělat efektivnější a intuitivnější. Autoři mj. také upozorňují na to, že adaptivní uživatelská rozhraní

zhoršují jejich zapamatovatelnost, matou uživatele a zhoršuje se ovladatelnost. To musí být vykompenzováno volbou obsahu, který je co nejvíce relevantní uživateli.

Aplikace sloužící jako studijní asistent by nejen měla pomáhat studentům s administrativními úkony, ale také pozitivně působit na formování dobrých studijních návyků. Stawarz, Cox a Blandford (2015) popisují studie a experimenty zaměřené na podporu návyků pomocí mobilních zařízení. Autoři popisují, že aplikace typu *reminder* sice podporují automatizování některých činností, nicméně lidé se stávají na těchto aplikacích závislí a při jejich absenci návyk rychle vymizí. Autoři doporučují následující pravidla pro podporu formování nových návyků s pomocí chytrých telefonů:

- Podpora vodítek – uživatel si může namísto času zvolit událost, na kterou naváže požadovaný úkol. Tím dojde ke snadnějšímu zapamatování i bez časové notifikace. Např. místo „v 13:00 zalít květiny“ si uživatel vytvoří poznámku „po obědě zalít květiny“.
- Notifikace by se měly používat pouze pro připomenutí, a to před definovaným vodítkem – související událostí.
- Vyvarovat se funkcím aplikace, které zvětšují závislost uživatelů na mobilních zařízeních.

Zpětná pozitivní vazba po splnění úkolu by také měla mít vliv na podporu formování návyků, nicméně autoři toto neprokázali, možná kvůli tomu, že zpětná vazba často působí jako nežádoucí.

Fridman, Weber, Greenstadt a Kam (2017) uvádí, jak lze na základě analýzy dat o chování uživatelů mobilních zařízení s velmi nízkou chybovostí (až 1 %) autentifikovat konkrétního uživatele, aniž by byla třeba autentifikační informace uložená v zařízení (Fridman, Weber, Greenstadt a Kam, 2017).

Tomitsch a kol. (2017) popisují, že uživatelé mají tendence projevovat hravé chování při interakci s displeji. Přidáním hravé funkcionality, která může být i skrytá, lze přilákat pozornost více uživatelů v okolí veřejného displeje (v našem případě displeje na chodbě nebo u učebny).

Hörold, Mayas a Krömker (2015) se zabývali správným umístěním a obsahem displejů ve veřejné městské dopravě. Displeje jsou umístěny tak, aby maximum přítomných cestujících mělo na displej dobrý výhled a sledování displeje nebylo namáhavé.

2.4 Lokalizace ve vnějších prostorách

Vzhledem k tomu, že cílová platforma mé práce je *iOS*, je třeba využít lokační API od společnosti *Apple* (Core Location, 2017). Tato služba využívá informace z mobilních sítí, Wi-Fi, GPS a Bluetooth. Zmíněné technologie jsou seřazeny od těch nejméně přesných po nejpřesnější. Služba *Core Location* automaticky vybírá technologie podle aktuální dostupnosti. Např. ve městech v husté zástavbě je GPS

signál slabý. V tom případě aplikace využívá okolních Wi-Fi přístupových míst pro upřesnění lokace. Přesnost lokalizace mobilních zařízení je však velmi náchylná na řadu faktorů, např. nevhodný obal zařízení, umístění poblíž jiných zařízení, materiál budov, vysoké hory nebo zástavba, nastavení času a časové zóny aj. Možnost získat lokaci je také omezena nastavením zařízení – uživatel může vypnout Wi-Fi, Bluetooth, zapnout režim v letadle nebo úplně zakázat lokační služby.

Aby aplikace mohla využívat lokační služby, je potřeba získat od uživatele souhlas. Na rozdíl od starších verzí platformy *Android* je na platformě *iOS* tento požadavek zobrazen až při spuštění aplikace, tedy až v okamžiku, kdy je funkce skutečně potřeba. Uživatel může toto nastavení kdykoliv změnit. Pro zpřesňování polohy se využívá tzv. *crowd-sourcing*: informace o okolních přístupových bodech Wi-Fi a pozice se odesílají na servery společnosti *Apple*, která poskytuje agregované informace dalším uživatelům. (About privacy and Location Services in iOS 8 and later, 2017)

Lokační API nabízí dvě možnosti získávání informací o poloze. Základní je získávání polohy pouze tehdy, když je aplikace na popředí. K tomu lze ještě povolit získávání polohy, i když je aplikace na pozadí. Nicméně kvůli šetření energie je tato možnost penalizována např. při schvalovacím procesu publikování aplikace v obchodu *AppStore*. (App Store Review Guidelines, 2017)

V rámci těchto služeb lze nastavit dvě klíčové vlastnosti – přesnost zjišťované lokace *desiredAccuracy* (několik kategorií v rozsahu metrů až kilometrů) a limit pro notifikování změn lokace *distanceFilter* (libovolné číslo). (Getting the User's Location, 2017).

Kromě standardní lokalizace pomocí *CoreLocation* frameworku existuje ještě služba *significant-change location*. Při zaregistrování této služby dojde k automatickému notifikování aplikace, jakmile se významně změní lokace (500 metrů a více, nelze specifikovat vývojářem). Výhodou této služby je, že aplikace je automaticky probuzena nebo spuštěna, pokud zrovna neběží na popředí. Tato služba je velmi úsporná. Nicméně vzhledem ke své nízké přesnosti by se mohla použít pouze na to, abych zjistil, zda uživatel přišel do areálu univerzity a v tom okamžiku se přepnul na přesnější lokační službu. Další úsporu energie lze docílit nastavením *pausesLocationUpdatesAutomatically* a *activityType*. V tom případě bude lokační služba zasílat informace o poloze jen tehdy, když je pravděpodobné, že se uživatel hýbe. (Using the Significant-Change Location Service, 2017; SwiftLocation, 2017)

Pro zjednodušení práce s lokačním API existuje řada knihoven. Např. *SwiftLocation* (2017) nebo *BBLocationManager* (2017) obaluje všechny běžné požadavky na lokační službu do elegantního kódu. *Ingeo SDK* (2017) automaticky upravuje parametry lokačního API s cílem minimalizovat vliv na baterii. Parametry stanovuje podle aktuální aktivity uživatele. Knihovna *SOMotionDetector* (2017) slouží pro rozpoznání aktivity uživatele (chůze, běh, jízda, počet kroků). Pro ladění funkcí spojených s lokací je velmi užitečný skript *set-simulator-location* (2017) umožňující nastavení simulované pozice přes příkazový řádek.

2.5 Lokalizace uvnitř budov

V této kapitole se zaměřím na *indoor* lokalizaci – tzn. uvnitř budov. V takových prostorách vede použití technologií zmíněných v předchozí sekci k velmi nepřesným výsledkům. K navigaci uvnitř budov je proto nutné použít nějaký lokální systém.

Různí autoři (Martin a kol., 2010; Li a kol., 2017) využívají ve své práci síť Wi-Fi, mobilní sítě a další senzory v zařízení pro získání přesné indoor lokace. Výhoda jejich práce je v tom, že není potřeba instalovat žádný další hardware uvnitř budovy. Nevýhoda spočívá v horší přesnosti, rozdílné přesnosti v různých podmínkách a složité prvotní konfiguraci.

Velmi experimentální a alternativní technologií je využití akustických neslyšitelných signálů pro triangulaci zařízení a určení jeho polohy. Nevýhodou je složitá instalace a nízká přesnost. (Ali, Javed, Hassanein a Oteafy, 2016)

Dalšími možnostmi lokalizace uvnitř budov je využití magnetického nebo visuálního trackování, které se používá ve virtuální realitě (Procházka, 2017). Tato řešení však vyžadují drahé vybavení jak na straně uživatele, tak v rámci místnosti. Slouží pro přesné určení pozice v místnosti. V rámci mé aplikace bude taková přesnost zbytečná a proto jsou tato řešení nevhodná.

V poslední době velmi populární standard je *iBeacon* vyvinutý společností *Apple* (iBeacon, 2017) a podobný standard *Eddystone* od společnosti *Google* (Beacons, 2017). Samotný hardware je vyvíjen a prodáván různými firmami (Estimote, 2017; Kontakt.io, 2017). Jedná se o malá zařízení s integrovaným úsporným *Bluetooth* vysílačem a malou baterií. Do prostoru vysílají svoje ID a příp. další informace. Jedná se především o tzv. *proximity* senzory – neslouží primárně pro přesné určení polohy, spíše pro detekci zařízení poblíž *beaconu*. Veškerá lokalizační logika a výpočty se odehrávají přímo v zařízení uživatele, které zachytí signál z beaconu. Díky své jednoduchosti a nízkým nákladům na pořízení je možné je nasadit ve velkém počtu i ve velkých budovách. (Zafari, Papapanagiotou, Devetsikiotis a Hacker, 2016)

Beacony typicky vysílají tři typy hodnot v nastaveném intervalu a nastavené intenzitě (Getting Started with iBeacon, 2017):

1. UUID – unikátní identifikátor aplikace,
2. *major value* – hodnota stanovená vývojářem, např. číslo podlaží,
3. *minor value* – hodnota stanovená vývojářem, např. číslo místnosti.

Signál zachycený mobilním zařízením je kategorizován do následujících zón pro účely přesnější lokalizace zařízení (Getting Started with iBeacon, 2017):

- *velmi blízká* – do 1 m,
- *blízká* – 1 až 3 m,
- *daleko* – více jak 3 m,
- *neznámá* – *beacon* nenalezen.

Beacony vysílají různé typy paketů popsané pomocí protokolů. Výše zmíněné údaje se týkají protokolu *iBeacon* od společnosti *Apple*. Např. beacony od společnosti *Estimote* vysílají kromě paketů ve formátu *iBeacon* také další pakety, např.:

- *Eddystone* UID, EID, URL, TLM – pakety ve standardu od společnosti *Google*,
- *Estimote Monitoring* – tento paket kombinuje *iBeacon* s informacemi o telemetrii *beaconu* a vylepšuje rychlost detekce,
- *Estimote Connection* – slouží pro konfiguraci beaconů,
- *Estimote Telemetry* – zasílání informací o stavu zařízení, pohybu, stavu baterie apod.,
- *Nearable* – pro malé přenosné beacony.

Jiní výrobci také nabízí různé formáty paketů. Některá zařízení podporují i vysílání více různých paketů současně, to však má negativní vliv na baterii a ve většině případů to není potřeba. (What is a beacon protocol?, 2017)

Pro účely mé práce využiji lokační službu založenou na kombinaci GPS, Wi-Fi a mobilních sítí pro detekci pozice ve venkovních prostorách. Pro určení pozice uvnitř budov budu využívat technologii *iBeacon*, která je vhodná pro můj záměr. Ostatní řešení, jako např. využití akustických signálů, jsou stále velmi experimentální a použitelná pouze v laboratorních podmínkách.

V rámci frameworku *Core Location*, používaném na platformě *iOS*, je nutné počítat s omezením pro monitorování oblastí. Lze zaregistrovat maximálně 20 oblastí. Toto omezení se týká služby, která hlídá výrazné změny lokace a probudí aplikaci, pokud změna nastane. Omezení lze obejít tím, že se dynamicky budou měnit monitorované regiony a automaticky se vyberou pouze ty nejbližší. (Region Monitoring and iBeacon, 2017)

Srovnání beaconů

V tabulce 1 jsem uvedl základní modely od největších výrobců *beaconů*. Vycházel jsem při tom ze seznamu největších výrobců beaconů (Vendors & Manufacturers of iBeacon or Bluetooth BLE Beacons, 2015). Informace o jednotlivých zařízeních jsou převzaty přímo z webových stránek výrobců.

Tabulka 1: Srovnání beaconů. Význam zkratek: P – senzor pohybu, T – senzor teploty, O – senzor ambientního osvětlení, M – magnetometr, L – tlakoměr, N – NFC, Ú – úložiště, V – venkovní použití, Z – zabezpečený mód přenosu, C – přizpůsobitelnost nebo možnost přidat senzory a hardware, S – automatické šetření baterie. (Accent Systems, 2017; BlueCats Beacons, 2017; Blue Sense Networks, 2017; Estimote, 2017; GCell iBeacon, 2017; Gimbal, 2017; Glimworm Beacons, 2017; Kontakt.io, 2017; Radius Networks, 2017)

Model	Životnost baterie v měsících (při intervalu)	Max. dosah	Nestandardní parametry	SDK od výrobce	Cena / ks
Accent Systems iBKS Plus	17 (100 ms)	70 m	V	ne	25 $
BlueCats AA Beacon	12 (100 ms)	100 m	V, Z	ano	30 $
BlueSense Beacon v3	15 (760 ms)	100 m	-	ano	21 $
Estimote Proximity Beacon	24 (950 ms)	70 m	P, T, S	ano	20 $
Estimote Location Beacon	60 (950 ms)	200 m	P, T, O, M, L, N, C, S	ano	33 $
GCell G300 Universal iBeacon	60 (100 ms)	25 m	V	ano	25 $
Gimbal Proximity Beacon Series 21	18 (100 ms)	50 m	V, Z	ano	30 $
Glimworm BATTERY+ BEACONS	1 (150 ms) nebo USB	50 m	C	ne	25 $
Kontakt.io Beacon	12 (100 ms)	70 m	Z, C	ano	20 $
Kontakt.io Beacon Pro X4	44 (100 ms)	70 m	Z, O, P, N, V, C, S	ano	28 $
Radius Networks RadBeacon X4	18 (100 ms)	50 m	V	ano	24 $

Seznam výrobců v tabulce 1 rozhodně není úplný. Mezi další výrobce patří např. *Fujitsu*, *Beaconinside*, *PayPal*, *Qualcomm* a mnozí další. Existuje také mnoho dalších výrobců, kteří neprodávají své beacony přímo, ale orientují se na velkokapacitní odběr B2B. Také samotné mobilní telefony mohou sloužit jako beacony (Danova, 2014). V případě speciálních požadavků, jako např. voděodolnost, prachuvzdornost, možnost připojení vlastního hardwaru, napájení ze sítě, napájení z USB portu, někteří výrobci nabízí i speciální modely s těmito vlastnostmi.

Kromě základních beaconů uvedených výše existují i další typy beaconů určené pro specifické nasazení:

- video beacony pro připojení k displeji: např. zatím neprodávaný *Estimote Video Beacon* napájený přes USB se zabudovaným Wi-Fi a 1 GB úložištěm (Estimote, 2017);
- solární beacony (GCell iBeacon, 2017);
- mini beacony, přívěsky (Gimbal, 2017; Estimote, 2017);
- brány, lokalizační beacony a centrální prvky (Kontakt.io, 2017).

Je nemožné jednoznačně vybrat nejlepší beacon. Mezi jednotlivými zařízeními jsou velmi malé rozdíly. Většina beaconů má k dispozici SDK. Cena jednotlivých beaconů je velmi podobná, při větším odběru jsou k dispozici množstevní slevy. Životnost baterie se pohybuje od několika měsíců po roky, závisí to především na typu použité baterie. Nejmenší životnost je u beaconů s baterií typu mince, nejdelší životnost je u beaconů používajících několik AA baterií. Životnost baterie je ovlivněna především vysílacím intervalem. Podle specifikace společnosti Apple je tento interval 100 ms, který je dostatečný i pro zachycení běžcem nebo autem. Ve většině případů však stačí nižší interval. Toho výrobci často využívají pro šetření energie. Další úspory lze dosáhnout pomocí různých šetřivých módů, které automaticky mění interval podle intenzity zařízení v okolí. Síla vysílaného signálu nemá na životnost takový vliv jako vysílací interval.

Vzhledem k podobným technickým parametrům beaconů mohou být klíčovým kritériem pro výběr také kvalitativní parametry, např. komplexnost SDK, dostupnost přehledné dokumentace, četnost aktualizací SDK, angažovanost vývojářů ve veřejných diskuzích, rychlost opravy nahlášených chyb, reference na reálné využití u velkých společností, podpora nových verzí mobilních platforem apod. Zmíněná kritéria je však obtížné vyčíslit a použít pro vzájemné porovnání beaconů. Např. četnost aktualizací SDK totiž může znamenat mnoho nových vylepšení, ale také opravy chyb. Tato kritéria jsou tedy vhodná spíše až pro finální rozhodnutí před větším nasazením beaconů.

Ve velkém srovnání beaconů (Danova, 2015) dosáhly nejlepšího hodnocení následující beacony. Jedná se však o dva roky starý test a proto některé parametry již nemusí být aktuální.

- *Estimote* – nejlepší designové provedení a uživatelská přívětivost,
- *Kontakt.io* – nejlepší výkon,
- *MPact* – vhodný pro velkoplošné nasazení,
- *Gimbal* – dobré zabezpečení.

Pro účely mé práce jsem vybral beacony od společnosti *Estimote*. Tyto beacony mají velmi propracované SDK a uživatelsky přívětivou konfigurační aplikaci. Komunita je rozsáhlá a vývojáři jsou aktivní na fórech. Mají také nejvíce zabudovaných

senzorů a nejvíce příležitostí pro různé způsoby využití. Jsou tedy dle mého názoru nejvhodnější pro prototypování nové aplikace.

Nasazení v praxi

Společnost *Locatify* uvádí následující poznatky z praktického nasazení (Beacon FAQ, 2015):

- V závislosti na nastavené intenzitě vysílaného signálu a intervalu dochází k vybíjení baterie. Reálná výdrž je v rámci měsíců až let, záleží na konkrétním zařízení.
- Konfigurace beaconů se provádí přes webové rozhraní a speciální mobilní aplikaci. Výchozí konfiguraci je možné objednat přímo u výrobců, to je užitečné zejména při větším nasazení.
- Nejlepší přesnost měření lokace je při vzdálenosti do jednoho metru. Od 2 do 7 metrů je přesnost s odchylkou asi 1,5 m, na větší vzdálenost již prakticky nelze lokalizovat uživatele a beacony pak slouží pouze jako senzory přítomnosti v oblasti.
- Detekce beaconu v mobilním zařízení může v nejhorším případě trvat až 6 vteřin. V místech, kde je rychlost detekce kritická (např. vstupní chodba), je výhodné umístit naráz více beaconů pro zkrácení detekčního času.
- Signál je nejčastěji blokován kovy, vodovodním vedením a lidmi. Beacony by se proto měly umístit do výšky asi 2,5 metru nad zemí, max. 4 metry.

Perfitt (2015) uvádí další rady založené na svých zkušenostech s nasazením beaconů:

- V předstihu vytvořit plán rozmístění a údržbu beaconů (přiřazení vysílaných hodnot, konfigurace, výměna baterie, kontrola provozu).
- Nenastavovat beacony na plný výkon. Vede to k velkým nepřesnostem.
- Neumístit více beaconů s různými vysílanými hodnotami blízko sebe (i z druhé strany zdi).
- Někteří výrobci nedodržují specifikace stanovené společností *Apple* s cílem šetřit energii. To však může vést k opakovanému vyvolání navázaných událostí i přesto, že je zařízení na místě.
- Ukradené beacony nebo beacony vysílající stejné hodnoty mohou způsobit vyvolání akce na nesprávném místě (bezpečnostní riziko).

Další doporučení uvádí Girish (2016):

- Vzdělávání uživatelů ohledně využívání beaconů (např. infografikou).

- Jasná a transparentní bezpečnostní politika, aby se uživatel nebál sdílet svoji lokaci.
- Informovat uživatele o příležitostech, o které přišel kvůli tomu, že nemá zapnutý *Bluetooth*.

2.6 Parametry cílových zařízení

Při návrhu mého řešení budu předpokládat následující cílová zařízení a požadavky.

Minimální verzi operačního systému v mobilním zařízení jsem stanovil na *iOS* verze 10, která je aktuálně (říjen 2017) přítomna v 89 % mobilních zařízeních od společnosti *Apple* (App Store - Support, 2017). V okamžiku zveřejnění mé aplikace bude již tato verze na naprosté většině používaných zařízení. Nejstarší zařízení podporující verzi 10 je *iPhone 5*, v případě tabletu je to *iPad* čtvrté generace (IOS 10, 2017). Kromě mobilního telefonu může uživatel vlastnit i libovolné chytré hodinky, které mu budou zobrazovat notifikace z mobilní aplikace.

Pro zobrazení informací budou kromě telefonů, tabletů a hodinek využity displeje na chodbách. Displeje jsou dvojího typu. První typ je černobílý s elektronickým inkoustem. Jedná se o řešení na míru. Obsah je zobrazován tak, že se do zařízení pošle rastrový obrázek, který se vykreslí. Vykreslování zajišťuje k displeji připojený minipočítač *Raspberry Pi* verze 3 připojený k síti. Tento typ displejů má úhlopříčku 13 palců a HD rozlišení. Druhým typem displejů jsou velké barevné displeje na chodbách, které jsou většinou běžnými televizemi s uhlopříčkou v řádech desítek palců a HD nebo Full HD rozlišením. Na těchto displejích bude obsah zobrazen jako webová stránka. Na televizích poběží internetový prohlížeč s vykreslovacím jádrem *WebKit*, např. *Google Chrome*. Pokud samotná televize neumožní zobrazení webové stránky, bude toto zajištěno připojením minipočítače *Raspberry Pi Zero W* přes HDMI kabel a napájením přes USB výstup televize. (interní zdroje)

2.7 Komunikace s displeji

Alt, Shirazi, Kubitza a Schmidt (2013) popisují různé způsoby přenosu obsahu mezi uživatelem a displejem. Popisují vytváření obsahu na mobilu a jeho přenos na displej a také naopak přenos obsahu z displeje do telefonu. Metody jsou seřazeny podle času nutného pro vykonání akce uživatelem. Následuje seznam základních způsobů předání informace mezi uživatelem a displejem:

1. automatická interakce bezdrátově s využitím mobilního telefonu (Bluetooth, NFC apod.),
2. přímý dotyk na displej,
3. explicitní akce uživatele (zadání kódu, vyfocení QR kódu, e-mail, SMS).

Vzhledem k povaze mé aplikace připadá v úvahu pouze bezdrátová a co nejvíce automatická komunikace. Jedině tímto způsobem bude naplněn požadavek na přirozené uživatelské rozhraní.

Existuje řada možností, jak technicky zajistit komunikaci mezi zařízením uživatele a displejem. V první řadě jsou to tzv. *video beacony*, o kterých jsem již psal v části *Lokalizace uvnitř budov*. Jedním ze zástupců video beaconů je *Estimote Mirror*. K displeji je připojen pomocí HDMI portu a napájen je ze sousedního USB. Jeho instalace je tedy velmi jednoduchá, protože není potřeba samostatné napájení ani baterie. Zařízení je vybaveno technologiemi *Bluetooth* a *Wi-Fi*. Dokáže přijímat signál z dalších beaconů, komunikovat s mobilními zařízeními a také s jakýmkoliv dalším *Bluetooth* hardwarem. Také dokáže komunikovat s libovolným serverem. Výrobce uvádí jako příklad využití zvýraznění času odletu na letištní tabuli podle uživatelů v okolí. A to je velmi podobné využití jako v případě mé práce. Zobrazování obsahu na displeji je velmi jednoduché, SDK má v sobě připravené základní bloky uživatelského rozhraní, které lze snadno použít a rozmístit na displeji. Tvorba UI je podobná tvorbě UI na mobilních telefonech, např. tvorbě UI pomocí knihovny *SnapKit* (Snapkit, 2017). Kromě toho lze využít SDK v jazyku *JavaScript* a napsat si vlastní uživatelské rozhraní displeje v HTML a *JavaScriptu*. Nahrání tohoto rozhraní se provádí připojením beaconu k počítači přes USB. Beacon je vybaven úložištěm o velikosti 1 GB. Bohužel v současné době (podzim 2017) není zatím tento beacon volně k prodeji. (Estimote, 2017; Cast content to Mirror from a mobile app, 2017; Build your own Mirror template, 2017)

Další možností komunikace je využití *Google Chromecast*. Pomocí *Cast SDK v2* je možné se připojit z mobilní aplikace k displeji a zobrazovat na displeji vlastní obsah podobně jako v případě *Estimote Mirror* (Remote Display | Cast, 2017). Podobná zařízení nabízí i další společnosti, např. společnost *Apple* s vlastním zařízením *Apple TV* (tvOS for Developers, 2017).

Problémem výše uvedených řešení je situace, kdy bude potřeba komunikovat se zařízeními více uživatelů a rozhodovat se, jaký obsah zobrazit. V takové situaci bude pravděpodobně třeba vytvořit řešení na míru. Může se jednat o minipočítač, jako je např. *Raspberry Pi* (Raspberry Pi, 2017) připojený k displeji. Dále bude potřeba server jako prostředník mezi mobilními zařízeními uživatelů a displeji. Zobrazení obsahu na displeji pak může probíhat třemi způsoby:

- Na minipočítači připojeným k displeji bude spuštěný webový prohlížeč zobrazující interaktivní webovou stránku komunikující se serverem (HTML a *JavaScript*). Alternativně může prohlížeč běžet přímo na displeji, pokud to displej/televize umožňuje.
- Na serveru se vykreslí rastrový obrázek, který se následně pošle do minipočítače, který zajistí vykreslení na připojeném displeji.
- Na minipočítači u displeje bude běžet jednoduchá grafická aplikace a API, do kterého bude server posílat požadavky na změnu obsahu. Aplikace může být

implementována např. pomocí *C++* a *Qt*. Tyto nástroje jsou vhodné pro vývoj nenáročných grafických aplikací pro vestavěná zařízení. (Qt, 2017).

2.8 Serverová služba

Při výběru prostředí pro implementaci serverové části mého systému jsem stanovil následující požadavky.

- *Rychlost* – Uživatelé budou aplikaci používat často za chůze a ve spěchu, je proto nezbytná velmi rychlá odezva systému.
- *Prototypování* – Navrhovaná aplikace je značně specifická a některé části jsou inovativní. Není možné dohledat zdokumentované metody implementace těchto částí, které by určovaly jednoznačně nejlepší způsob implementace. Zvolené vývojové prostředí by proto mělo umožnit snadné prototypování. Díky snadnému prototypování bude možné rychle vyzkoušet a otestovat různé varianty implementace a vybrat z nich tu nejlepší.
- *Podpora specifických funkcí* – Prostředí by mělo umožňovat vykreslování obrázků a také podporu pro vytvoření rozhraní nutného pro komunikaci s dipleji a mobilními zařízeními.
- *Budoucí vývoj a rozšiřitelnost* – Vytvořený systém bude provozován na univerzitě a je velmi pravděpodobné, že do jeho vývoje budou v budoucnu zapojeni i další lidé. Zvolené vývojové prostředí tedy musí mít strmou křivku učení a programovací jazyk musí být dostatečně obecný a běžně používaný. Zvolené prostředí také musí mít dostatečnou základnu tvůrců a dostatečnou dokumentaci, aby bylo zajištěno, že v následujících letech nedojde k jeho úpadku.

Vzhledem k výše uvedeným požadavkům budou pro implementaci serverové aplikace vhodné dynamicky typované jazyky se slabou typovou kontrolou jako je např. *PHP*, *Python* nebo *JavaScript (Node.js)*. Všechny tyto jazyky, na webových serverech běžně používané, obsahují potřebné knihovny, např. pro vykreslování obsahu do obrázku: *PHP* obsahuje knihovnu *GD* (PHP: GD - Manual, 2017), *Node.js* obsahuje podobnou knihovnu *node-gd* (2018) apod. Pro tvorbu komunikačního rozhraní existuje také řada knihoven (Top 12 Best PHP RESTful Micro Frameworks (Pro/Con), 2015). Běžně používané webové frameworky, jako je *Nette* nebo *Laravel*, obsahují nástroje pro tvorbu REST rozhraní (Nette – jak vytvořit REST API – json, 2014; Otwell, 2017). V *Node.js* je tvorba REST rozhraní zcela přirozená díky použití *JavaScriptu*. Všechny zmíněné jazyky jsou rovněž dlouhou dobu vyvíjeny a udržovány a mají velmi dobrou uživatelskou základnu a podporu.

Vzhledem k požadavku na rychlost a požadavku na snadné prototypování jsem pro svoji práci vybral prostředí *Node.js*, které dle testů vyniká oproti ostatním v rychlosti díky asynchronnímu zpracování a škálovatelnosti (Kazankov, 2017; Sanchez, 2018). Zároveň je dostatečně flexibilní a efektivní pro rychlý vývoj prototypů.

To zejména díky konsistenci použitého jazyka na frontendu i backendu, velmi dynamické povaze *Javascriptu*, snadnému rozšiřování pomocí mnoha dostupných knihoven a značné volnosti při psaní kódu (Farias, 2017).

Javascript v rámci *Node.js* vyniká v rychlosti vyřizování požadavků oproti ostatním dynamicky typovaným jazykům, nepřekoná však jazyky jako je *C++* nebo *Go* (Sanchez, 2018). Pro uvažované nasazení systému by však měl být *Node.js* postačující.

Pro ukládání dat lze použít různé databázové technologie. Mohou to být klasické SQL databáze (např. *MySQL*) nebo alternativní NoSQL databáze (např. *MongoDB*, *LokiJS* nebo *Microsoft Document DB*). Pro účely mé práce zvolím databázi *LokiJS* (2018). Tuto databázi jsem zvolil na základě řady vlastností vhodných pro moji práci. Vycházel jsem při tom ze srovnání Buckler (2015).

- Vzhledem k experimentální povaze aplikace nelze předem přesně definovat podobu dat. V serverové aplikaci bude podstatou spíše rychlé procházení připravených dat než složité dotazy. Proto bude NoSQL databáze vhodnější než SQL databáze.
- Data v databázi *LokiJS* jsou ukládány ve formátu JSON a bude tak možné je přímo zpracovávat jako objekty v serverovém prostředí *Javascriptu* bez nutnosti konverze. Manipulace s daty bude velmi snadná.
- Databáze *LokiJS* je typu in-memory a dosahuje podle výkonnostních testů velmi dobrých výsledků. Zároveň podporuje i automatickou persistenci.
- Snadná bude i komunikace s displeji pomocí REST rozhraní. Data budou v databázi, na serveru i při komunikaci ve stejném formátu.
- Rozhraní databáze je velmi jednoduché, databáze je nenáročná na prostředky a konfiguraci, ale přitom umožňuje všechny běžné operace jako je indexace, filtrování a řazení.

2.9 Vývojové prostředí

V této sekci jsou uvedeny základní nástroje potřebné pro vývoj jednotlivých částí navrhovaného systému.

Design – Návrh uživatelského rozhraní bude vytvořen v programu *Sketch* (The digital design toolkit, 2017) a *Affinity Designer* (Affinity, 2017). Jsou to moderní a často používané nástroje pro zvolené platformy.

Mobilní aplikace – Pro vývoj mobilní aplikace použiji vývojové prostředí *Xcode* verze 9 (Xcode, 2017) a programovací jazyk *Swift* verze 4 (Swift, 2017). Jedná se o jediné úplné vývojové prostředí pro vývoj na platformu *iOS*.

Server – Serverová aplikace bude vyvíjena v jazyku *JavaScript* v rámci *Node.js* ve vývojovém prostředí *PhpStorm* (PhpStorm, 2017). Serverová aplikace bude běžet na serveru s operačním systémem *Linux*.

Pokročilé displeje – Zobrazovací aplikace pro velké barevné displeje budou napsány jako webové stránky v rámci prostředí *PhpStorm* (PhpStorm, 2017).

Jednoduché displeje – Pro vývoj nízkoúrovňové zobrazovací aplikace pro minipočítač *Raspberry Pi* připojený k e-inkovým displejům byl použit jazyk *Python*. Tato aplikace není součástí této práce, protože byla vyvíjena nezávisle na této práci.

Beacony – Pro konfiguraci beaconů se vždy používá aplikace přímo od výrobce a není potřeba žádné speciální programové vybavení.

3 Metodika

3.1 Uživatel a jeho potřeby

Definice uživatele

Podle metodiky návrhu uživatelského rozhraní zaměřeného na uživatele (Tidwell, 2011; Procházka, 2017) nejprve definuji uživatele a jeho potřeby. Využiji k tomu tzv. persony a vytvořím fiktivní osoby reprezentující typické uživatele.

Jana je novou studentkou prvního ročníku manažersko-ekonomického oboru. Má okolo sebe spoustu neznámého – nezná umístění učeben, kanceláří, menz, nechápe kreditový systém a ještě navíc ji na první přednášce z podnikové ekonomiky zavalili spoustou nové látky a tím pádem nic nestíhá. Do školy chodí každý den, má nabitý rozvrh a bývá ve škole většinou od půl deváté, dvakrát týdně přichází do školy už před sedmou hodinou ráno. Ze školy odchází v různou dobu během odpoledne.

Pavel je studentem informatického oboru, studuje třetí ročník bakalářského studia. Již má dostudované téměř všechny předměty a je velmi zběhlý v řešení administrativních povinností. Aktuálně se naplno věnuje psaní své závěrečné práce. Do školy chodí akorát dvakrát týdně, dojíždí vždy ráno mezi 8:45 – 9:00.

Michaela je aktivní studentkou druhého ročníku oboru Finance. Již je zběhlá v řešení běžných studijních povinností. Je pilnou studentkou, má dostudováno většinu předmětů a ráda by příští rok vyjela do zahraniční, nebo by si našla praxi ve firmě. Míšu zajímá vše ohledně dění v akademickém životě, je členkou akademického senátu, účastní se dobrovolných přednášek a konferencí. Začíná pomalu přemýšlet o tématu své bakalářské práce. Do školy chodí nepravidelně i několikrát během dne, protože bydlí 5 minut od školy a může si kdykoliv v pauze zajít domů.

Pozorování uživatelů

Na základě vlastního pozorování jsem shromáždil následující údaje o pohybu lidí na frekventovaných místech v budově Q Mendelovy univerzity.

- *Hlavní vchod do budovy*: Co se týká příchodů a odchodů ze školy, platí, že největší četnost lidí u vstupu je čtvrt hodiny před začátkem přednášek ve dnech pondělí až středa. Z budovy odchází nejvíce lidí kolem 11. hodiny na oběd a dále nárazově po skončení přednášek ve velkých přednáškových místnostech. Experimentálně bylo zjištěno, že od okamžiku, kdy je displej pro uživatele poprvé viditelný (uživatel vstoupil do budovy) a mezi jeho průchodem pod displejem (displej již je mimo zorné pole uživatele), uplyne 5–10 vteřin. Odhaduji, že nejméně polovina této doby je potřeba, aby uživatel stihl změnu obsahu zaregistrovat a vnímat jeho podstatu. Je proto nutné identifikovat uživatele a adaptovat obsah na displeji za dobu kratší jak 2,5 vteřiny.
- *Studijní oddělení*: Na chodbě na studijním oddělení je nejvíce lidí v prvních dvou týdnech semestru. V této době čekají studenti až několik desítek minut ve

frontě. Většinou zde neprobíhá komunikace mezi studenty, vzájemně se většinou neznají.

- *Chodby u učeben*: Na chodbách u učeben je nejvíce studentů před začátkem výuky, nejvíce přichází v deseti minutách před začátkem výuky. Na rozdíl od studijního oddělení zde probíhá živá diskuze.

Uživatelská funkční specifikace

V souladu s metodikou návrhu uživatelských rozhraní (Tidwell, 2011) nejprve navrhnu funkce na vysoké (uživatelské) úrovni. Teprve později je převedu na implementační úroveň. Mohu definovat 4 obecné cíle vytvářeného systému:

- Vyřešit problém, který tíží studenta.
- Předat studentovi řešení problému, o kterém sice student zatím neví, ale může ho pravděpodobně v blízké době řešit.
- Pomoci studentům hlídat termíny a hlídat plnění administrativních úkolů.
- Informovat uživatele a probudit v něm zájem o další informace.

Na základě cílů pak definuji základní funkce systému z pohledu uživatele:

- Sdělit uživateli primární obsah (např: upozornění na termín, řešení studijní povinnosti, pozvánka na událost, informace o probíhající výuce, fotky, videa, program konference, navigační informace, vlastní upomínka apod.).
- Sdělit uživateli na požádání více informací k primárnímu obsahu.
- Umožnit uživateli reagovat na primární obsah (např. komentář, označení jako to se mi líbí, označení úkolu jako vyřešeného, navigace na místo apod.).
- Umožnit uživateli uložit si obsah k přečtení na později.
- Notifikovat uživatele na primární obsah (na základě jeho lokace nebo času).

Implementační funkční specifikace

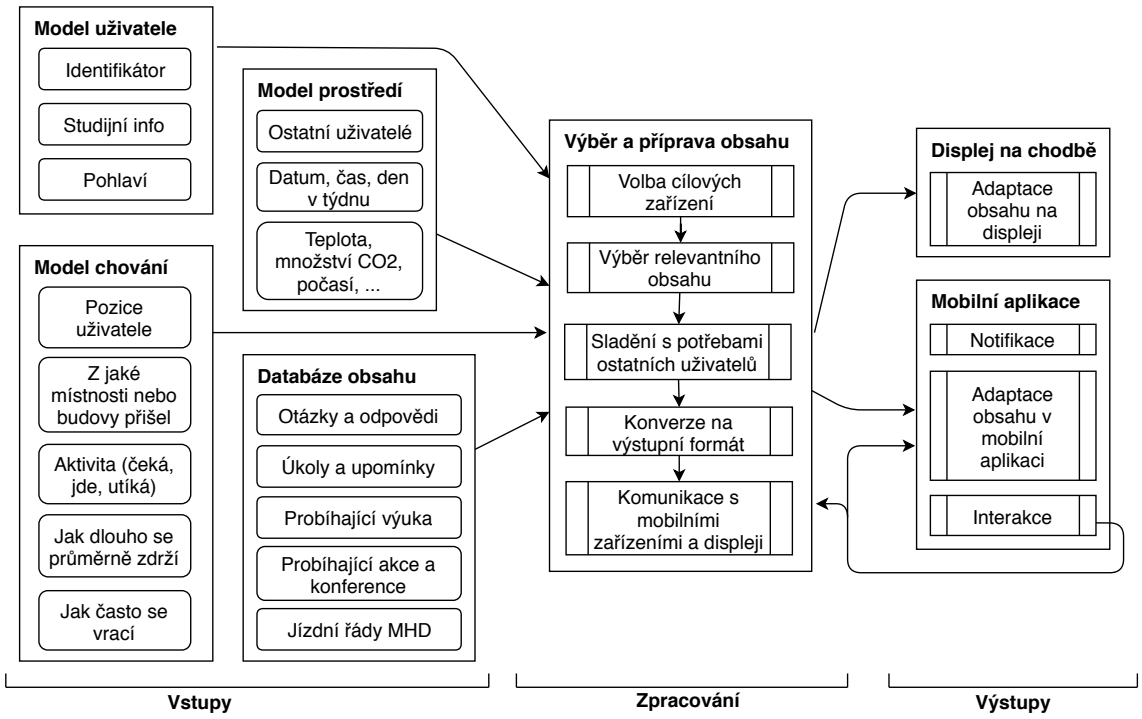
Funkční specifikaci navrhovaného systému z předchozí sekce dále převedu do implementační úrovně.

- Identifikovat uživatele.
- Identifikovat chování uživatele.
- Předat informaci směrem od zařízení uživatele k displeji.
- Předat informaci směrem od displeje k zařízení uživatele.
- Zobrazit relevantní obsah na displeji.

- Zobrazit relevantní obsah v mobilní aplikaci.
- Vyvolat notifikaci v mobilním zařízení.
- Reagovat na obsah v mobilní aplikaci.
- Uložit obsah k pozdějšímu přečtení v mobilní aplikaci.
- Zobrazit obsah, který byl uložen k pozdějšímu přečtení v mobilní aplikaci.
- Konfigurovat uživatelský profil v mobilní aplikaci.

3.2 Model systému

Navrhovaný systém definuji jako proces, který má určité vstupy, vnitřní logiku a výstupy. Následuje definice jednotlivých částí modelu. Celý model je znázorněn na obrázku 2. Při tvorbě modelů jsem vycházel z teoretických koncepcí autorů Vossen a kol. (2017), které jsem rozšířil, upřesnil a aplikoval do kontextu univerzity.



Obrázek 2: Komplexní model navrhovaného systému.

Model uživatele

Model uživatele zahrnuje *identifikátor uživatele*, pomocí kterého se bude identifikovat konkrétní uživatel. Tento identifikátor bude sloužit k jednoznačnému přiřazení obsahu k uživateli a k ukládání historických dat uživatele. Dalšími atributy tohoto

modelu jsou *studijní informace* zahrnující studijní ročník, obor studia, typ studia (bakalářské, navazující), aktuální rozvrh apod. V budoucnu bude možné v případě potřeby přidat další údaje jako např. pohlaví nebo věk.

Model chování

Model chování se tvoří pro konkrétního uživatele a obsahuje informace o chování uživatele: aktuální *pozice* uživatele (v případě indoor pozice je to místnost), *minulá pozice* (popř. místnost), aktuální *aktivita* (čeká, jde, utíká). Dále jsou v tomto modelu historické informace vztahující se k aktuálnímu místu, a to *průměrný čas* strávený na místě a *četnost návštěv* za týden.

V případě aktuální aktivity je nutné vzít v potaz i aktivitu v nedávné době, tedy v rámci posledních několika minut. Tedy i v případě, že aktuálně uživatel stojí na místě, může tento atribut nabývat hodnotu *utíká*, pokud v předchozích pěti minutách převažoval tento pohyb. Toto rozlišení je důležité pro správné filtrování obsahu. Spěchající uživatel ocení stručnější formu obsahu než uživatel čekající dlouho na místě.

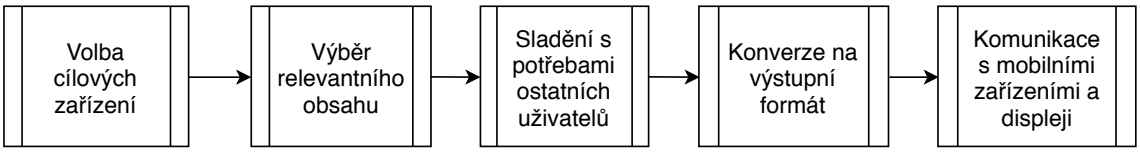
Model prostředí

Model prostředí zde představuje parametry okolí uživatele. Jedná se především o *modely ostatních uživatelů* ve stejné oblasti, *čas*, *datum* a *den v týdnu*. V budoucnu bude možné v případě potřeby rozšířit tento model o informace o *počasí*, teplotu v místnostech, stav oxidu uhličitého v přednáškových sálech a studovnách apod. Oblastí je zde myšlena aktuální místnost, popř. prostor kolem uživatele v dohledu na displej – tedy okruh 10 metrů.

Databáze obsahu

V databázi je umístěn veškerý obsah. Tento obsah může být tvořen otázkami a odpověďmi souvisejícími se studiem, upomínkami uživatelů, rozvrhy, informacemi o konferencích, jízdními řády apod.

Obsah může mít primární nebo sekundární podobu (i obě naráz). Primární obsah je zobrazován na displejích. Sekundární obsah pak představuje dodatečné informace zobrazitelné na požádání v mobilních zařízeních. Každá položka je kategorizována tak, že jsou jí přiřazeny konkrétní hodnoty atributů v modelu uživatele, modelu chování a modelu prostředí. Obsah pak může mít různé formy pro zobrazení na různých cílových zařízeních (např. jednu formu pro e-inkové displeje, druhou pro barevné).



Obrázek 3: Postup adaptace obsahu. Na počátku je určeno, která zařízení se budou aktualizovat. Dále je vybrán relevantní obsah z databáze na základě vstupů systému, které byly popsány v části Model systému. Poté dojde ke sladění potřeb uživatelů ve stejné oblasti. Obsah určený jako kandidát na zobrazení přejde do fáze přizpůsobení zobrazovacím možnostem cílového zařízení. Na závěr se obsah odešle do zařízení.

3.3 Adaptace obsahu

Volba cílových zařízení

Každá položka v databázi obsahu bude obsahovat výčet displejů a zařízení, na které je možné položku zobrazit. Proces aktualizace obsahu bude spouštěn dvěma způsoby:

a) Periodicky se každou minutu provede aktualizace obsahu pro všechny displeje. V tomto případě budou další fáze iterovány pro všechny displeje.

b) Na základě události (čas, změna lokace) je spuštěna aktualizace pro konkrétní displej nebo mobilní zařízení.

Výběr relevantního obsahu

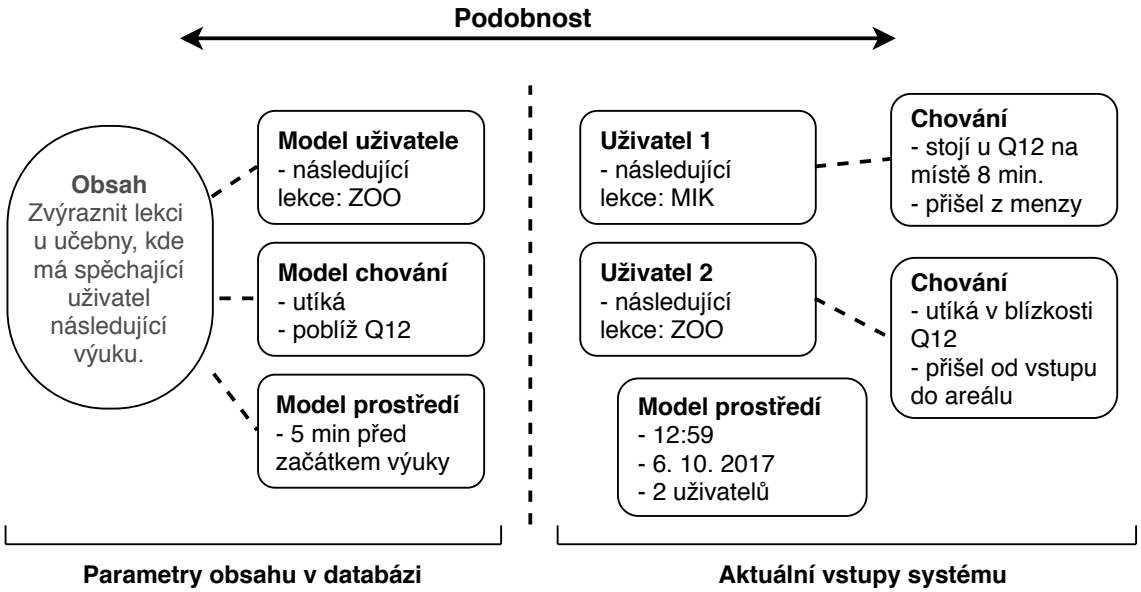
Každá položka v databázi je kategorizována tak, že je jí přiřazen model uživatele, prostředí a chování. Při výběru relevantního obsahu se porovnávají tyto modely s aktuálním modelem prostředí, modely všech přítomných uživatelů a modely interakce přítomných uživatelů. Pokud některý obsah nezávisí na všech modelech, je tento model jednoduše prázdný nebo určitý atribut není uveden. Konkrétní příklad je na obrázku 4.

Sladění potřeb všech uživatelů v okolí

V praxi bude často nastávat situace, kdy bude v blízkosti displeje více uživatelů. Jako příklad mohu uvést situaci u hlavního vchodu do budovy krátce po příjezdu šaliny[1] před devátou hodinou ráno. V tom případě obdržíme z předchozího kroku různý obsah pro různé přítomné uživatele. Vzhledem k omezené zobrazovací ploše displejů bude třeba rozhodnout, který obsah zobrazit.

Tento problém vyřeším zavedením několika atributů, které mohou být nastaveny u obsahu v databázi. První atribut `paralelní` nabývá hodnot pravda/nepravda a vyjadřuje, zda daný obsah může být zobrazen zároveň s dalším obsahem podporujícím paralelní zobrazení. Druhý atribut `priorita` nabývá hodnot z rozsahu přirozených čísel a umožňuje přímo upřednostnit důležitý obsah, jako je např. učebna u spě-

[1] Vzhledem k místu vzniku této práce bude dopravní prostředek tramvaj nazýván šalina.



Obrázek 4: Příklad výběru relevantního obsahu. Dojde ke shodě mezi obsahem v databázi (nalevo) a uživatelem 2 (napravo). Obsah je zde tvořen akcí – zvýrazněním lekce na displeji. Příklad konkrétní situace: Uživatelé si často nepamatují rozvrh a neví, ve které místnosti mají následující výuku. Uživatel 2 vstoupil do areálu univerzity a velkou rychlostí se přiblížil k místnosti Q12. Systém v tomto případě detekuje spěchajícího uživatele a zvýrazní mu lekci na displeji u učebny, kde má uživatel další výuku. Díky tomu se uživatel nemusí zdržovat čtením rozvrhu učebny nebo vytahováním telefonu.

chajícího studenta nebo důležité upozornění od vedení školy. Požadované vlastnosti uživatelů lze zabalit do atributů `anyUser` a `mostUsers` podle toho, zda požadujeme, aby dané vlastnosti splňoval alespoň jeden nebo nadpoloviční většina uživatelů. Navíc lze ještě pomocí atributu `prioritizeUserMajority` upřednostnit obsah, jehož atributy se shodují s nejvíce uživateli v okolí. Jinými slovy, podle počtu shodujících se uživatelů se přiměřeně navyšuje priorita obsahu.

V případě, že první položka s nejvyšší prioritou nebude podporovat paralelní zobrazení, další položky v pořadí se na displeji nezobrazí. Zobrazí se pouze v mobilních zařízeních, pokud to daný obsah umožní. V případě, že na prvních místech bude souvislá řada položek umožňující paralelní zobrazení, budou na displeji zobrazeny všechny tyto položky. Pokud tato souvislá řada bude přerušena položkou neumožňující paralelní zobrazení, nebude už tato ani další položky v pořadí zobrazeny.

Konverze na výstupní formát

Pokud přijde do této fáze více položek se stejnou prioritou, nebo více položek umožňující paralelní zobrazení a zároveň cílové zařízení neumožňuje paralelně zobrazit tolik položek, pak jsou zobrazené položky vybírány náhodně. Interval střídání je dán v atributech položky, nebo vypočítán automaticky na základě velikosti zob-

razovací plochy a množství textu k přečtení. Stanovení minimální doby, po kterou bude položka zobrazena, je velmi důležité, jinak by mohlo při větší frekvenci procházejících uživatelů docházet k rychlému střídání obsahu a žádný uživatel by nestihl svůj obsah přečíst.

Vzhledem k různým velikostem výstupních zařízení je třeba obsah vhodně adaptovat, aby byl dobře čitelný na všech zařízeních. Také je potřeba různým způsobem navodit u uživatelů možnosti interakce. Pro každý typ cílového zařízení a pro každý typ obsahu bude připravena šablona. Připravím dva základní typy šablon – webovou a rastrovou. V případě webové šablony bude tato šablona nachystána na cílovém zařízení a do zařízení se v závěrečné fázi přípravy obsahu pošlou pouze data, kterými se přímo na straně klienta v zařízení šablona vyplní. V případě rastrové šablony je obsah přidán do šablony přímo na serveru a do cílového zařízení je poslán obsah i s grafickým rozhraním jako jeden rastrový obrázek. Pro různé typy obsahu a různá zařízení budou připraveny tyto šablony:

- velká webová (pro velký displej na chodbě),
- mobilní webová (pro zobrazení obsahu v mobilní aplikaci),
- malá rastrová (pro malé displeje, např. e-ink displeje u učeben).

Komunikace s výstupními zařízeními

Aktualizace obsahu na displejích umožňující zobrazení webové stránky bude probíhat pomocí zprávy zaslané ze serveru do displejů prostřednictvím technologie *web sockets*. K odeslání dojde jen pokud server identifikuje obsah jako nový nebo změněný – tedy odlišný od současně zobrazeného obsahu na displeji.

V případě e-inkových displejů bude komunikace probíhat obdobně. Rozdíl bude akorát ve formátu přenášené zprávy. V případě těchto displejů bude odeslán obrázek na REST rozhraní minipočítače připojeného k displeji.

V případě mobilní aplikace se obsah vyžádá požadavkem z mobilní aplikace na server. Ke stažení dojde při spuštění aplikace uživatelem, případně automaticky na pozadí po změně lokace.

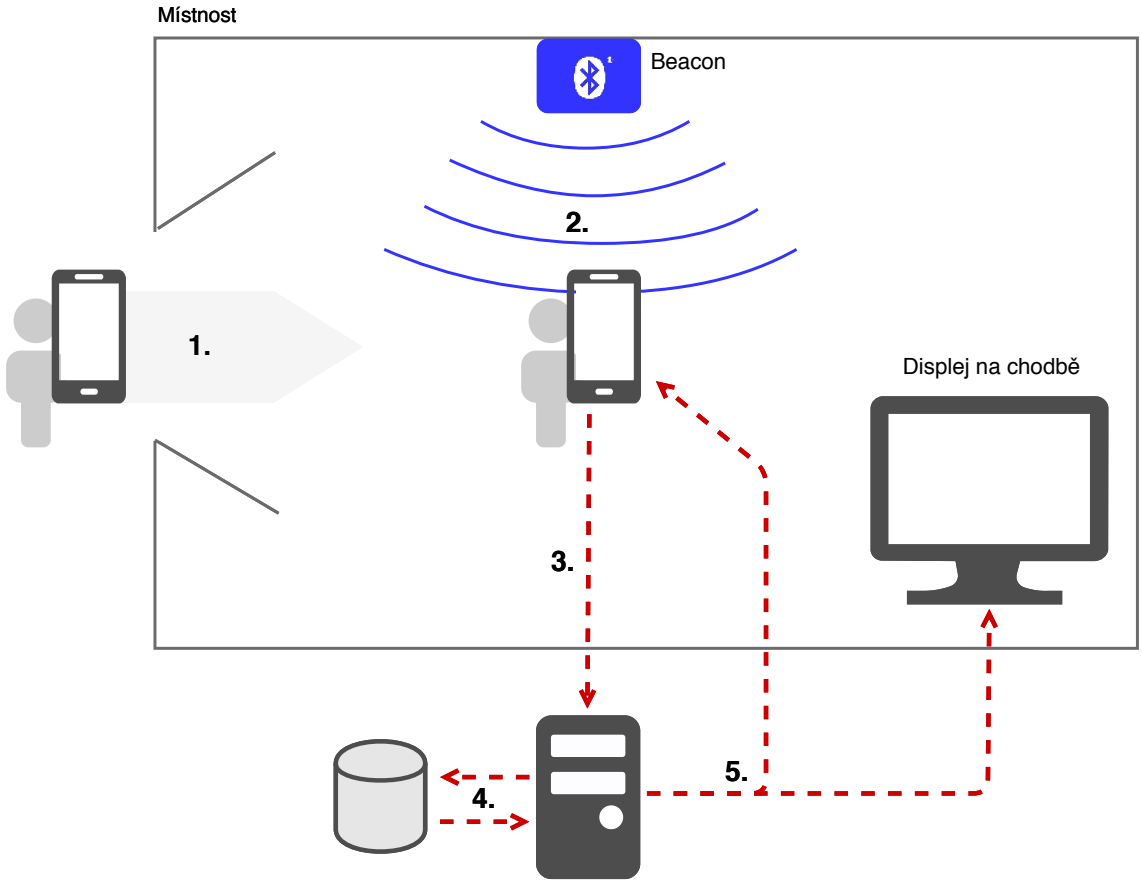
3.4 Identifikace uživatele

Aby bylo možné přizpůsobovat obsah displeje na chodbě podle potřeb uživatelů přítomných v blízkosti displeje, je třeba uživatele v okolí displeje identifikovat. Identifikace bude probíhat následovně:

1. Uživatel má při sobě chytrý telefon se zapnutým *Bluetooth*. Vstoupí do oblasti s displejem.
2. Chytrý telefon uživatele zachytí signál z beaconu, který je v místnosti nainstalován.

3. Signál z beaconu obsahuje identifikátor místnosti. Aplikace v chytrém telefonu podle toho pozná, ve které místnosti se uživatel nachází.
4. Chytrý telefon odešle na server identifikační údaje uživatele spolu s lokací uživatele a další data o chování uživatele (viz vstupy systému v sekci 3.2 Model systému).
5. Server vytvoří dotaz do databáze obsahu a vybere obsah, který je nejvíce relevantní k uživatelům v dané oblasti (viz sekce 3.3 Adaptace obsahu).
6. Vybraný relevantní obsah je odeslán na cílový displej nebo zařízení uživatele.

Celý proces je znázorněn na obrázku 5.



Obrázek 5: Obrázek zobrazuje rozpoznání uživatele a následné přizpůsobení obsahu na displeji podle potřeb uživatele. Modře je znázorněn Bluetooth signál z beaconu. Červené šipky představují komunikaci mezi zařízeními. 1. Uživatel vstoupí do místnosti. 2. Signál z beaconu je zachycen mobilním zařízením. 3. Zařízení odešle informaci o lokaci a identifikaci uživatele na server. 4. Server vybere relevantní obsah v databázi. 5. Vybraný obsah se zobrazí na displeji na chodbě nebo v zařízení uživatele.

3.5 Interakce

S obsahem na displeji bude uživatel interagovat prostřednictvím svého mobilního zařízení. Ke každé položce obsahu bude možné přiřadit akce ve formě odkazů na webové stránky nebo do jiných aplikací. Dodatečný obsah a interaktivní akce budou zobrazeny pouze v mobilních zařízeních. Na displejích bude třeba umístit vodítko naznačující možnost využití mobilní aplikace.

Jako příklad uvedu zobrazení textu „Conference ECE 2018“ na displeji a možnost zobrazit si podrobný program konference na svém zařízení. V takovém případě mobilní aplikace odešle požadavek na server. Server vrátí aktuální obsah relevantní pro danou oblast a uživatele. Konkrétně v tomto příkladu uživatel obdrží podrobný program konference. Uživatel si může zobrazený obsah uložit do svého zařízení k přečtení na později, ohodnotit konferenci na sociální síti nebo se třeba přepnout do navigace a najít cestu do ubytovacího zařízení.

3.6 Testování

Mobilní aplikace bude otestována pomocí jednotkových testů, které ověří funkčnost tříd – zejména práci s daty, lokálním úložištěm, synchronizaci apod. Pro mobilní aplikaci budou také napsány akceptační testy, které zajistí automatické otestování uživatelského rozhraní.

Také serverová služba a aplikace pro displeje budou otestovány pomocí jednotkových, integračních a akceptačních testů. Jednotkové a integrační testy ověří funkčnost jednotlivých komponent serveru, zejména se budou týkat zpracování dat a práce s databází. Akceptační testy serveru budou zaměřeny na správnou reakci serveru. Pro každý test budou na server nejprve zaslány testovací atributy prostředí. Poté dojde k ověření, zda server správně zareagoval a zobrazil očekávaný obsah na správných displejích. Akceptační testování nebude probíhat na reálných zařízeních, displeje budou pro zjednodušení testování simulovány. V rámci akceptačních testů budu vycházet z případů užití uvedených dále.

Na závěr bude provedeno uživatelské testování na skupině dobrovolníků zaměřené na použitelnost a přívětivost uživatelského rozhraní displejů a mobilní aplikace. Bude připraveno několik scénářů, které budou testovací uživatelé provádět. Bude sledováno, zda při plnění zadaného úkolu narazí uživatelé na problémy a jak rychle dokáží zadaný úkol splnit. Také bude vyhodnocen celkový dojem uživatelů vzhledem k jejich reálným problémům.

3.7 Konkrétní případy užití

V této části navrhnu několik případů využití systému. Tyto návrhy budou dále přeneseny do databáze obsahu a bude na nich celý systém otestován.

Displej u vstupu

- Zobrazit následující lekci a učebnu pro spěchajícího studenta.
- Při probíhající konferenci zobrazit možnost navigace do jiné budovy, do menzy, na zastávku MHD apod. Na displeji je pouze možnost, samotná navigace probíhá v zařízení uživatele.
- Pro odcházející uživatele zobrazit nejbližší odjezdy MHD. Nabídnout možnost navigace v mobilním zařízení.

Displej u učebny nebo přednáškového sálu

- Pro spěchajícího studenta zvýraznit displej u cílové učebny, název předmětu a čas do začátku výuky.
- Pro čekající studenty zobrazit informace o aktuální a následující výuce.
- Při konferenci zobrazit program.
- Při zrušení výuky zobrazit tuto informaci a možnost navigace do jiné učebny.

Displej na studijním oddělení

- Zobrazit studijní otázky a odpovědi z brožury *Průvodce prváka*. Relevantní otázky a odpovědi budou vybírány podle studijních parametrů uživatele a parametrů okolí.
- Umožnit studijní referentce upřednostnit obsah pro urychlení vyřizování fronty. Např. zobrazit informaci o tom, co si mají studenti připravit pro vyřízení potvrzení o studiu.
- Zobrazit zajímavost o přítomných uživatelích: pokud bude v blízkosti displeje většina uživatelů s danou vlastností, zobrazit tuto informaci.

Jakýkoliv displej

- Vedení školy nebo studijní referentka může přidat důležité upozornění, např. informaci o toaletách mimo provoz nebo informaci o termínu pro odevzdání závěrečné práce cílenou na studenty posledního ročníku.
- Uživatel si může přidat notifikaci vázanou na vodítko, např. „po škole dokončit projekt“ nebo „až přijdu do školy, zajít na studijní oddělení“. Na displeji, a příp. i v mobilním zařízení, je zobrazeno upozornění, jakmile nastane definovaný čas nebo uživatel přijde na definovanou lokaci.

4 Vlastní práce

Popis implementace jsem rozdělil do tří částí. První se věnuje serverové službě poskytující obsah. Druhá část se zabývá zobrazením obsahu na displejích. Poslední část se věnuje mobilní aplikaci – zobrazování obsahu, interakci s obsahem a detekci aktivity uživatele s využitím lokačních služeb.

Celý systém jsem navrhl a implementoval tak, aby byl maximálně škálovatelný a rozšiřitelný. Rovněž jsem kladl důraz na snadnou budoucí údržbu. Systém je proto složen z mnoha malých komponent, které jsou vnitřně velmi primitivní a zcela na sobě nezávislé. Případná úprava nebo rozšíření je tak velmi jednoduché. Změny v jedné komponentě neovlivní jiné komponenty. Použité vývojové prostředí je velmi snadné na naučení a přístupné i pro vývojáře bez předchozích zkušeností. Systém byl navržen pro univerzální použití a není pevně svázán s obsahem pro prostředí univerzity. Je tak možné velmi snadno systém přizpůsobit různému využití i mimo univerzitu.

4.1 Serverová služba

Architektura

Serverová služba pro poskytování obsahu byla implementována v prostředí *Node.js*. Struktura služby odpovídá komplexnímu modelu, který byl navržen v kapitole Metodika. Pro vytvoření rozhraní serverové služby jsem použil jednoduchý framework *Express* (2018). Základní komponenty tvořící serverovou službu popíši na následujícím příkladu. Komponenty jsou také znázorněny na obrázku 6.

1. Uživatel se přiblíží k displeji. Z jeho mobilního zařízení odchází data na server pomocí POST požadavku na adresu `/user-data`.
2. Hlavní *controller* přijme tato data a uloží je do databáze pomocí třídy `Users`, která se stará o sledování přítomných uživatelů a monitorování jejich aktivity. Např. nastavuje typ aktivity "čekání" v případě, že uživatel je na místě více jak 5 minut, nebo uživatele odstraní v případě, že nepřišla nová data od uživatele za poslední půl hodinu.
3. Na základě přijetí dat od uživatele (příp. samostatným GET požadavkem na adresu `/update-displays`) je *controllerem* spuštěn proces výběru obsahu. Tento proces je zajištěn třídou `DisplayUpdate`.
4. Pro každý cílový displej je provedena následující série kroků.
    a) Třída `RelevantContents` zajistí vybrání relevantního obsahu z databáze pomocí informací získaných přes třídu `Environment`. Algoritmus výběru relevantního obsahu bude popsán dále.

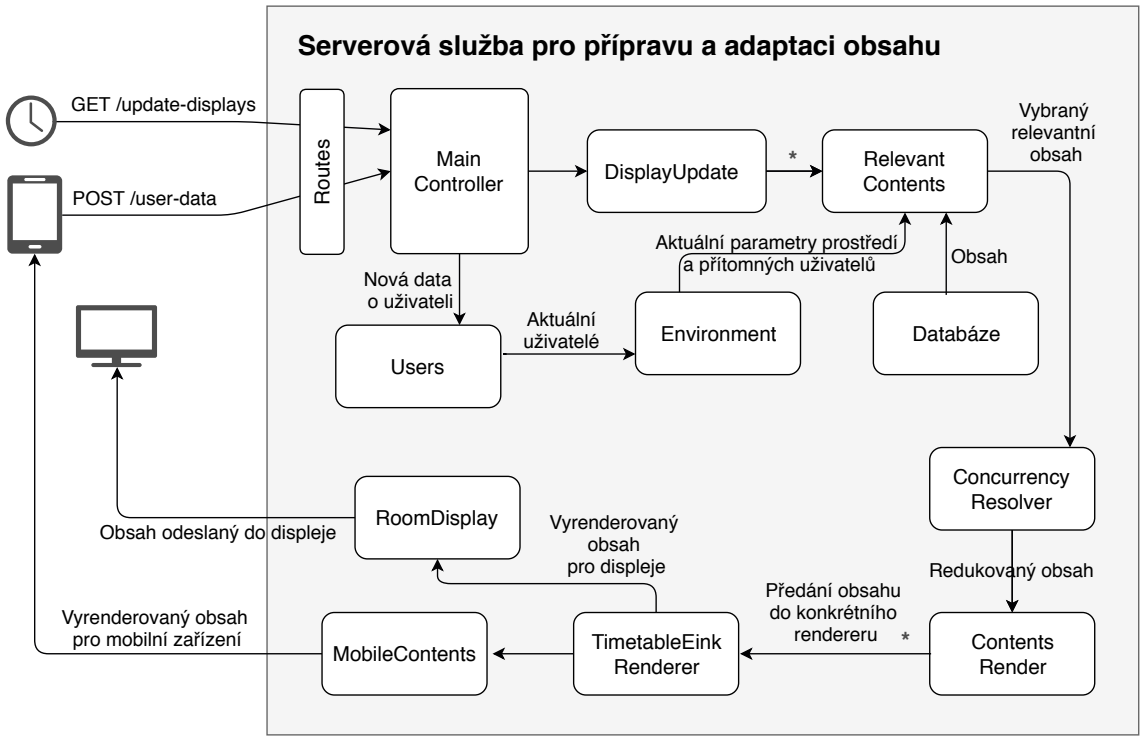
b) Třída `ConcurencyResolver` vyřeší konkurenci podle návrhu popsaného v Metodice v případě, že v předchozí fázi bylo vybráno více položek obsahu se stejnou prioritou.

c) Třída `ContentsRender` zajistí vykreslení vybraného obsahu pomocí připravených tzv. *rendererů* – tříd připravujících obsah pro konkrétní displeje, např. zajistí načtení rozvrhů ze speciální databáze a vykreslí data do PNG obrázku, příp. připraví data do formátu JSON. Příkladem rendereru je `TimetableEinkRenderer`, který zajišťuje vykreslení rozvrhu do rastrové podoby pro e-inkové displeje. Jiný příklad je `AlertWebRenderer`, který vykresluje upozornění ve formátu pro barevné webové displeje.

d) Jakmile jsou všechny položky vyrenderovány, je obsah odeslán do displeje. Každý typ displeje má svoji třídu, která poskytuje informace o displeji a zajišťuje také odesílání obsahu do displeje, např. třída `EinkDisplay` nebo `LargeDisplay`.

5. Kromě odeslání do displejů je vybraný obsah také odeslán do mobilního zařízení uživatele. Obsah je odeslán jako odpověď na požadavek ze zařízení uživatele (viz bod 1). Přípravu mobilního obsahu zajišťuje třída `MobileContents`.

Serverová služba obsahuje také další pomocné komponenty, které však nejsou přímou součástí z důvodu udržení co největší obecnosti systému. Mezi tyto komponenty patří např. import studijního obsahu do databáze, nástroj pro stahování jízdních řádů, import rozvrhů nebo systém pro správu obsahu.



Obrázek 6: Základní komponenty serverové služby. Šipky znázorňují tok dat mezi komponentami při přijetí dat od uživatele a následné aktualizaci obsahu na zařízeních. Hvězdička za *DisplayUpdater* naznačuje iteraci pro všechny cílové displeje, hvězdička za *ContentRender* naznačuje iteraci pro všechny vybrané položky obsahu. Na místě *TimetableEinkRenderer* a *RoomDisplay* může být jiná třída v závislosti na konkrétním obsahu a cílovém zařízení.

Algoritmus pro výběr obsahu

Při výběru obsahu z databáze dochází k porovnávání atributů obsahu v databázi s aktuálními daty o prostředí a přítomných uživatelích. Princip algoritmu vysvětlím opět na příkladech.

Následuje ukázka parametrů obsahu, který zobrazuje rozvrhy na displejích (viz zdrojový kód 1). Na řádcích 2 až 6 je definice požadovaných parametrů. Jedná se o parametry prostředí: dny v týdnu a časový rozsah – nemá význam zobrazovat rozvrhy o víkendech, kdy neprobíhá běžná výuka. Podívejme se také na další parametry. Parametr *renderer* určuje vykreslení pomocí rendereru „timetable“, který vykresluje obsah jako rozvrhy. Parametr *displays* určuje, na kterých displejích je možné obsah zobrazit. Parametr *priority* nastavuje standardní prioritu nula, protože se jedná o běžné sdělení. Parametr *parallel* určuje, zda daný obsah je možné zobrazit paralelně s jiným obsahem na jednom displeji.

Zdrojový kód 1: Obsah v databázi reprezentující zobrazení rozvrhů.

```
1  {
2    env: {
3      daysInWeek: [1, 2, 3, 4, 5],
4      timeFrom: '06:00:00',
5      timeTo: '19:00:00'
6    },
7    renderer: 'timetable',
8    displays: ['entrance1', 'q12', 'q11', 'entrance2'],
9    priority: 0,
10   parallel: true
11 }
```

Další příklad je komplexnější (viz zdrojový kód 2). Jedná se o upomínku pro studenta, který neuhradil poplatek za studium. Na řádku 2 až 12 je definice požadovaných parametrů. Parametry *timeFrom* a *timeTo* určují, kdy se má obsah zobrazovat. Parametry, které jsou uvedeny pod *anyUser* musí být splněny alespoň jedním uživatelem v okolí displeje. Zde konkrétně je vyžadován alespoň jeden uživatel s identifikátorem 1, jehož aktivita je „pohyb“ a poslední lokace uživatele, na které zůstal déle jak 5 minut, je mimo areál univerzity (nebo-li právě přišel do školy). Mezi další parametry patří opět *renderer* určující vykreslení pomocí rendereru *alert*, který vykresluje obsah jako výstrahu. Parametr *displays* určuje, na kterých diplejích je možné obsah zobrazit. Parametr *priority* nastavuje velmi vysokou prioritu, protože se jedná o důležité sdělení. Parametr *parallel* určuje, že daný obsah není možné zobrazit paralelně s jiným obsahem na jednom displeji. Parametr *data* a *mobileData* jsou pak hodnoty, které jsou při vykreslování v *rendereru* předány do vykreslovacích šablon.

Zdrojový kód 2: Obsah v databázi reprezentující upomínku studenta.

```
{
  env: {
    timeFrom: '06:00:00',
    timeTo: '12:00:00',
    anyUser: {
      behaviour: {
        activity: 'moving',
        lastStayLocation: 'out-of-campus'
      },
      userId: 1
    }
  },
  renderer: 'alert',
  displays: ['entrance1'],
  priority: 100,
  parallel: false,
  data: {
    title: 'UPOMINKA',
    text: 'Stale jste neuhradil poplatek za studium!',
    icon: 'user.png'
  },
  mobileData: {
    title: 'UPOMINKA',
    text: 'Uhradte prosim poplatek za studium co nejdrive. Pri prodleni
     Vam hrozi az vylouceni ze studia. Vice informaci na odkazech nize.
    ',
    links: [
      {
        url: 'https://mendelu.cz/studijni-oddeleni',
        title: 'Uredni hodiny'
      },
      {
        url: 'http://mendelu.cz/studijni-rad',
        title: 'Studijni rad'
      }
    ]
  }
}
```

Poslední příklad reprezentuje studijní otázku a odpověď cílenou na studenty prvních dvou semestrů informatických oborů (viz zdrojový kód 3). Obor studia a semestr je uveden jako pole hodnot – v tomto případě algoritmus vybere obsah, pokud se shoduje alespoň jedna z uvedených hodnot s aktuální hodnotou uživatelů. Díky atributu `prioritizeUserMajority` se tato položka upřednostní před ostatními, pokud bude nejvíce uživatelů v okolí displeje mít shodu právě s touto položkou.

Zdrojový kód 3: Obsah v databázi reprezentující studijní otázku a odpověď.

```
{
  env: {
    anyUser: {
      study: ['Ekonomicka informatika', 'Automatizace rizeni a informatika'],
      semester: [1, 2]
    }
  },
  prioritizeUserMajority: true,
  renderer: 'faq',
  data: {
    icon: 'question',
    title: 'Jak se dostat do laboratore?',
    text: 'Podle oblasti zajmu (mobilni aplikace, virtualni realita, robotika) se muzete zapojit do tymové práce na skupinovem projektu. Navic budete odmeneni ve forme stipendia.',
    highlighted: false
  },
  mobileData: {
    title: 'Jak se dostat do laboratore?',
    text: 'Podle oblasti zajmu (mobilni aplikace, virtualni realita, robotika) se muzete zapojit do tymové práce na skupinovem projektu. Navic budete odmeneni ve forme stipendia. Pro vice informaci a kontakty se podivejte na nase stranky nize.',
    links: [
      {
        url: 'https://spatialhub.mendelu.cz/',
        title: 'Laborator virtualni relaity'
      },
      {
        url: 'http://aistorm.mendelu.cz/',
        title: 'Roboticka laborator'
      }
    ]
  },
  displays: ['studydepartment'],
  priority: 0,
  parallel: true
}
```

Výstupem z tohoto algoritmu (*ContentResolver*) jsou relevantní položky, u každé je také uvedena priorita, která je příp. navýšena o počet shod s uživateli v případě využití parametru `prioritizeUserMajority`. Na základě této priority je

obsah seřazen a předán do další fáze – komponenty *ConcurrencyResolver* popsané dříve.

4.2 Displeje

Uživatelské rozhraní

Pro každý testovací případ užití uvedený v Metodice jsem navrhl uživatelské rozhraní. Před samotným návrhem jsem pozoroval displeje, které vidíme dennodenně ve svém okolí. Při pozorování jsem se snažil identifikovat základní problémy těchto displejů. Ukázky špatně navrženého rozhraní jsou na obrázku 7: v levé horní části je displej zobrazující pozvánku na kurz genetiky. Problémem je velké množství textu, špatné rozmístění a mezery, zbytečné údaje. Vpravo nahoře je displej v autobusu zobrazující následující zastávky a konečnou stanici. Ačkoliv je tento displej větší a podporuje více barev a větší rozlišení než displej na spodním obrázku, uživatelské rozhraní není dle mého názoru správně vytvořeno. Kvůli špatně použitým barvám, množství malého textu a špatnému uspořádání je tento displej špatně čitelný z větší vzdálenosti a velmi nepřehledný. Pro srovnání na spodním obrázku je jednořádkový displej, kde je stejný typ obsahu zobrazen v mnohem více konzistentní a čitelnější podobě, která není matoucí ani při rychlém pohledu bez zvýšené kognitivní zátěže.

Podobně jsem se také pokusil charakterizovat správné vlastnosti displejů, viz obrázek 8. Na základě těchto pozorování jsem si stanovil několik zásad:

- Aby byl text na displejích čitelný i z velké vzdálenosti, je zapotřebí použít vhodné písmo. Na základě diskuze o písmech vhodných pro čtení z velké vzdálenosti (Fonts that can be read at a distance, 2011) jsem vybral font *Roadgeek* (Adams, 2005). Podobné typy písma, jako je *Roadgeek*, jsou používány na dopravním značení u silnic. Lepší čitelnosti je dosaženo díky větším mezerám mezi jednotlivými znaky i mezi křivkami uvnitř znaků. Nedochází tedy k takovému splývání při čtení z větší vzdálenosti.
- Na základě inspirace tabulemi odletů na letištích a návěstí nad dálnicemi (viz obrázek 8) jsem došel k závěru, že je třeba co nejvíce omezit grafické prvky. Pro oddělení jednotlivých zobrazených informací je většinou postačující mezera. Linie a bloky zabírají zbytečné místo a snižují přehlednost při čtení z větší vzdálenosti.
- Většina pozorovaných panelů a displejů má v praxi černé nebo tmavě modré pozadí. Bílý text na tmavém pozadí zajistí dostatečný kontrast pro dobrou čitelnost. Zároveň nebude docházet k nepříjemnému oslnění, které by mohlo způsobovat bílé pozadí s černým textem. To vychází z předpokladu, že displej bude umístěn na chodbě, kde je po většinu doby šero. V případě budoucího umístění displeje do venkovních prostor lze uvažovat denní a noční variantu zobrazení s různým pozadím.

Na základě těchto zásad a obecných zásad tvorby uživatelských rozhraní popsaných v rešerši jsem navrhl uživatelské rozhraní pro různé typy displejů a různé typy obsahu, viz obrázky 9–13.

Na obrázku 9 nalevo je zobrazení rozvrhu na e-inkovém černobílém displeji u učebny. Na rozdíl od klasických tištěných rozvrhů je zde vždy výrazně uvedena nejbližší výuka. Naspodu je pak další výuka během aktuálního dne. Toto uspořádání vychází z faktu, že většinu studentů v blízkosti displeje zajímá aktuální výuka. Výuku v další dny lze zobrazit v mobilní aplikaci. Stačí, když uživatel spustí aplikaci a na základě lokace se automaticky zobrazí kompletní rozvrh. Dostupnost rozšiřujícího obsahu přes mobilní aplikaci indikuje vodítko – malá ikonka v pravém dolním rohu displeje. Na obrázku 9 napravo je znázorněno, jak lze podobným způsobem zobrazovat program konference v dané místnosti.

Zobrazení nejbližší výuky na velkém barevném displeji u vstupu do budovy lze vidět na obrázku 10. U každé lekce je uvedena učebna, poschodí a směr, kudy se k učebně vydat. Naspodu je pak odpočet doby do začátku výuky. Navíc modrou barvou jsou zvýrazněny lekce pro studenty, kteří aktuálně prochází v blízkosti displeje. Červeně se zvýrazní lekce v případě změny učebny. Na pravém obrázku je opět seznam nejbližších lekcí, navíc je ještě paralelně zobrazena uživatelská notifikace.

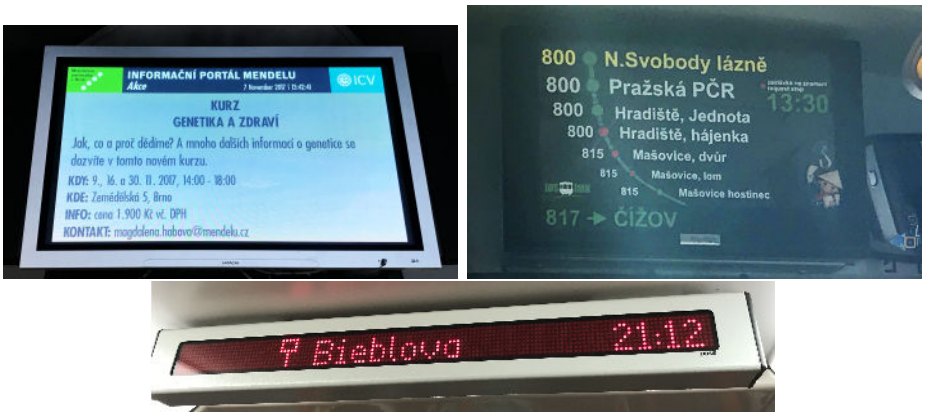
V případě konání konference lze displeje efektivně využít pro uvítání návštěvníků (obrázek 11 nalevo). Napravo je pak podobný displej u východu zobrazující informace o večeři v rámci konference. Uživatelé si mohou zobrazit další informace o konferenci přes mobilní aplikaci nebo se přepnout do navigace, která je zavede do jiné budovy nebo třeba k východu z areálu, pokud již jsou běžné východy večer uzavřeny. Důležitou funkcí je opět spojení displejů s mobilními zařízeními. Obsah je v mobilní aplikaci adaptován okamžitě po přiblížení se k displeji.

Na obrázku 12 nalevo je displej určený pro studijní oddělení. Jsou zde zobrazovány otázky a odpovědi související s řešením častých studijních problémů. Otázky jsou zobrazovány tak, aby respektovaly potřeby většiny uživatelů aktuálně čekajících v blízkosti displeje. Pokud má uživatel o otázku zájem, lze si zobrazit další informace v mobilní aplikaci, kde se automaticky načtou aktuální otázky právě zobrazené na displeji. U otázky mohou být i tlačítka, např. tlačítko pro vytvoření upozornění k zobrazenému úkolu. Jak můžete vidět na levém obrázku nahoře, jedna otázka je modře zvýrazněna. Jedná se o otázku, která byla připnuta studijní referentkou. Kromě studijních informací se na displejích mohou zobrazovat i zábavné informace dynamicky generované na základě přítomných uživatelů, jak lze vidět na obrázku 12 napravo.

Na obrázku 13 lze vidět displej, který zobrazuje výstrahu. Výstraha může být samozřejmě zobrazena pořád, nicméně s využitím vlastností mého frameworku lze výstrahu cílit pouze na konkrétní skupinu uživatelů, příp. na konkrétní uživatele. Např. upozornit studenty posledního ročníku na uzávěrku závěrečných prací nebo výrazně upozornit nezodpovědného studenta na zaplacení poplatku za studium.

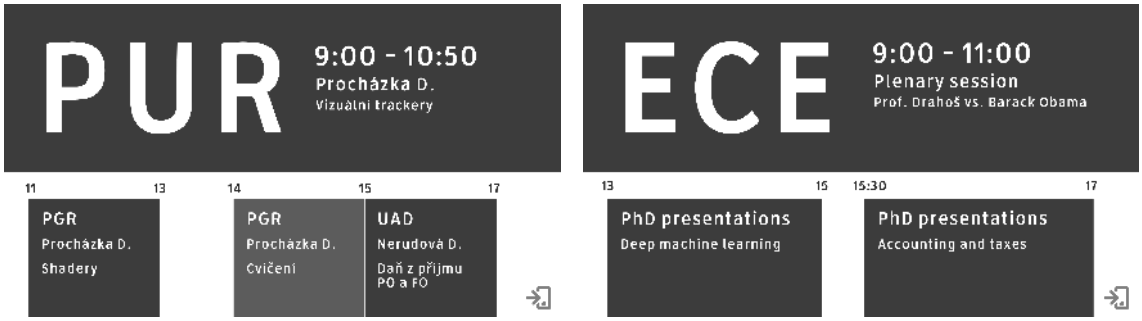
Nejbližší odjezdy MHD lze vidět na displeji u východu z budovy (obrázek 13 napravo). Opět lze využít mobilní aplikaci, kde ihned po přiblížení se k displeji

budou další informace nebo třeba tlačítko pro navigaci na zastávku.

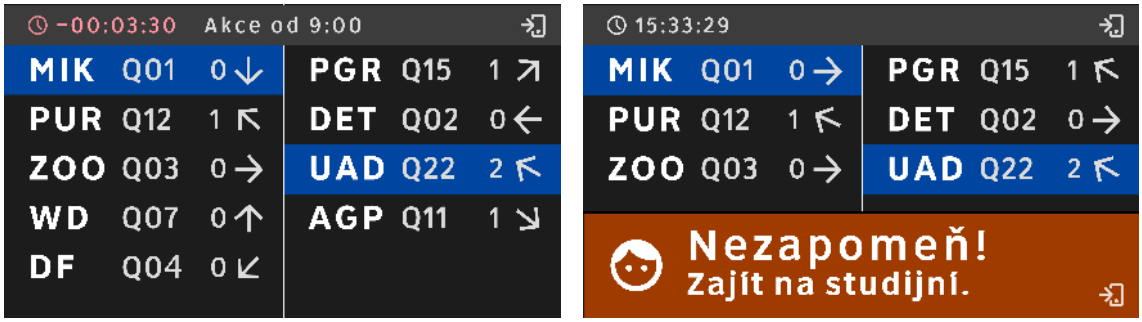


Obrázek 7: Pozorování současných displejů na univerzitě a v dopravních prostředcích. Levý obrázek vyfocen na MENDELU, ostatní v dopravních prostředcích IDS JMK.

Obrázek 8: Při návrhu uživatelského rozhraní jsem se inspiroval tabulemi s odlety na letištích a dopravním značením u silnic. Levý obrázek: Bahnfrend (Own work) [CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0)], via Wikimedia Commons; pravý obrázek: Jacqueline Macou [CC0].

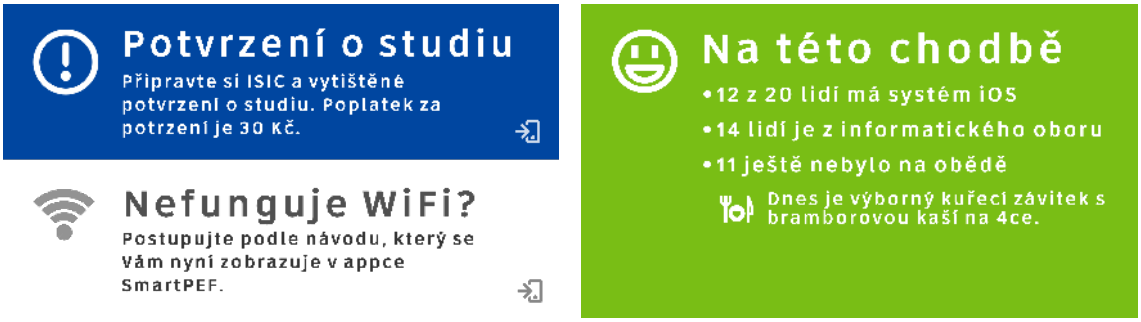


Obrázek 9: Zobrazení rozvrhu na e-inkovém černobílém displeji u učebny (nalevo). Zobrazení programu konference (napravo). Pro zobrazení rozvrhu na další dny lze využít mobilní aplikaci, kde se automaticky stáhne kompletní rozvrh nebo program.

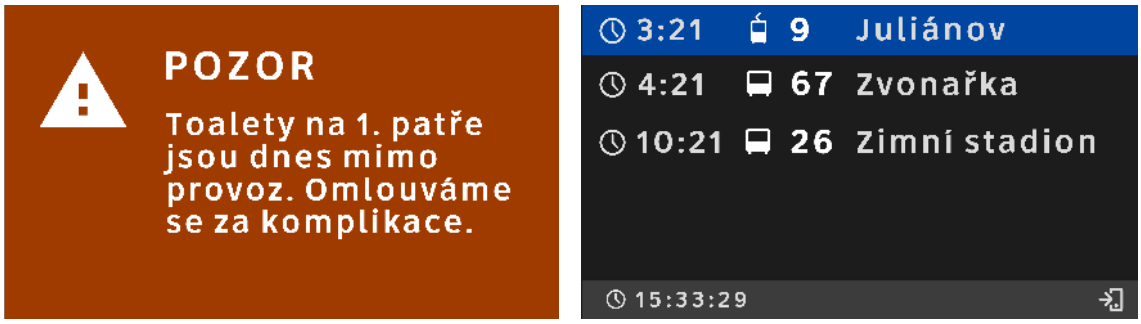


Obrázek 10: Zobrazení nejbližší výuky na velkém barevném displeji u vstupu do budovy. Modře zvýrazněny lekce pro konkrétní uživatele v okolí displeje. Na pravém obrázku je opět seznam nejbližších lekcí, navíc je ještě paralelně zobrazena uživatelská notifikace.

Obrázek 11: Nalevo: uvítání návštěvníků konference, napravo: navigace na slavnostní večeři.



Obrázek 12: Nalevo je displej určený pro studijní oddělení s častými otázkami a odpověďmi. Otázky respektují potřeby uživatelů v okolí displeje. Otázky a související odkazy lze zobrazit přes mobilní apikaci. Napravo je obrazovka se zábavnými informacemi o přítomných uživatelích.



Obrázek 13: Nalevo je obrazovka s výstrahou. Napravo je obrazovka u východu z budovy zobrazující nejbližší odjezdy MHD. Opět lze využít mobilní aplikaci, např. pro navigaci na zastávku.

Řídící aplikace

E-inkové displeje s připojeným minipočítačem *Raspberry Pi* mají vlastní REST rozhraní pro přijímání obrázků ze serveru. Minipočítač zajistí vykreslení přijatého obrázku na displeji. Implementace této části je součástí jiné práce (Ing. Vít Ondroušek, Ph.D.) a nebudu ji zde více popisovat.

Velké barevné displeje (televize) zobrazují webovou aplikaci `web-display`, která zajišťuje přijímání obsahu ze serveru pomocí technologie *web sockets*. Přijatý obsah je zobrazován pomocí *iframů* s využitím šablon popsaných dále. Tato řídící aplikace automaticky rozpoznává typ přijatého obsahu a přizpůsobuje zobrazení. V případě paralelního obsahu zobrazí více *iframů* pod sebou s různým obsahem. Aplikace také rozpoznává změny. To znamená, že v případě nového obsahu přidá *iframe*, v případě nepřijetí nových dat k již zobrazenému obsahu odstraní související *iframe* a v případě přijetí dat k již zobrazenému obsahu pouze obnoví data v *iframu*. Všechny změny jsou doprovázeny animací. Díky použití *iframů* může mít každý typ obsahu nezávislou šablonu v podobě jednoduché webové stránky. Editace i vytvoření šablony je tak velmi jednoduché a rychlé i pro méně zkušeného uživatele. Jak již bylo uvedeno na začátku kapitoly, celý systém jsem se snažil vytvořit jako množinu malých primitivních a nezávislých komponent.

Alternativním přístupem pro implementaci zobrazovací aplikace by bylo použití frontendového frameworku typu *Angular.js*. Toto řešení jsem však po otestování zavrhl z důvodu zbytečné složitosti frontendu a těžké budoucí udržitelnosti. Mé řešení pomocí *iframů* a nezávislých šablon je obecnější a bude do budoucna vhodnější.

Šablony

Pro velké displeje umožňující zobrazení webové stránky jsou připraveny *webové šablony*, které se zobrazují pomocí webové aplikace popsané v předchozí sekci. Pro černobílé e-inkové displeje jsou připraveny *vykreslovací šablony*. Tyto šablony převezmou data a vykreslí je společně s grafickými elementy do obrázku ve formátu PNG.

Pro každý typ obsahu může být připraveno i více šablon. Každá šablona má na serveru svůj *renderer* (vykreslovač). Podle typu cílového displeje se zvolí správná varianta vykreslovače a šablony. Např. pro rozvrh jsem připravil vykreslovač `TimetableWebRenderer` napojený na webovou šablonu `timetable` a vykreslovač `TimetableEinkRrenderer` napojený na šablonu pro vytvoření PNG obrázku. Podobně jako pro rozvrh jsem vytvořil šablony pro všechny typy obsahu podle uživatelského rozhraní popsaného na začátku této kapitoly.

4.3 Mobilní aplikace

Uživatelské rozhraní

V souladu s principy tvorby uživatelských rozhraní popsaných v rešerši jsem navrhl uživatelské rozhraní také pro mobilní zařízení. Na obrázku 14 je základní obrazovka s výpisem obsahu. Pro co největší zjednodušení navigace jsem základní obrazovku udělal jako jeden dlouhý seznam. Díky tomu není nutné řešit navigaci mezi různými obrazovkami, ale vše je umístěné v rámci jedné obrazovky. Pro navigaci mezi jednotlivými typy obsahu tak stačí skrolovat. Následuje popis jednotlivých částí této obrazovky.

První sekce se jmenuje „Poblíž“ a obsahuje položky související s aktuální lokací uživatele. Je to tedy např. obsah z displeje poblíž uživatele. U každé položky je název, místo a čas vzniku události, textový popis, odkazy na webové stránky, do jiných částí aplikace nebo do jiných aplikací.

Ve funkčních požadavcích bylo stanoveno, že uživatel má mít možnost si uložit položku na přečtení později. I tuto funkcionalitu jsem integroval přímo do hlavního seznamu. Místo klasického tlačítka „Uložit do oblíbených“ a seznamu uložených položek je u každé položky tlačítko s ikonkou připínáčku. Po kliknutí na toto tlačítko se změní pozadí tlačítka a jeho natočení. Připnutá položka zůstane v seznamu natrvalo, dokud uživatel položku opět neodepne pomocí stejného tlačítka. Připnuté položky jsou v seznamu hned za sekcí „Nejblíž“, takže nebude problém pro uživatele tyto položky později dohledat. Skrolování dolů je typická akce, kterou uživatel provádí, pokud nemůže něco najít.

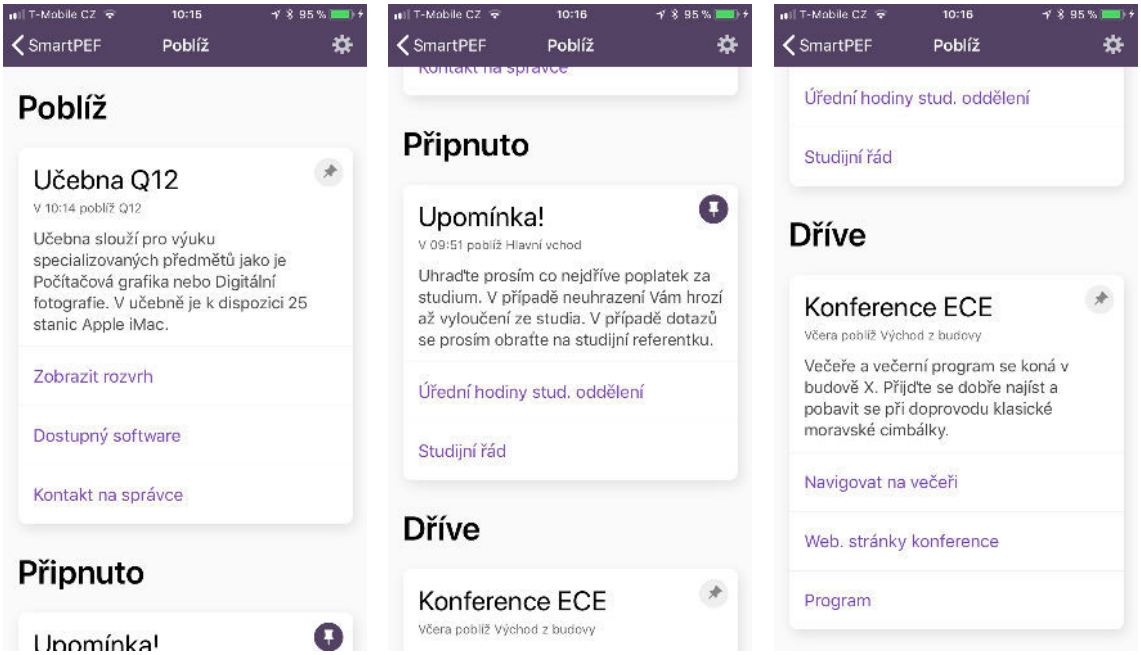
Poslední sekce na hlavní obrazovce je nazvaná „Dříve“ a obsahuje starší položky. Automaticky jsou promazávány položky starší jak 14 dní. Samozřejmě jsou promazávány pouze nepřipnuté položky. Do této sekce se přesouvají položky ze sekce „Nejblíž“ při každé změně lokace. Nicméně pouze ty položky, které si uživatel v aplikaci zobrazil nebo na které nějak zareagoval, např. pomocí kliknutí na notifikaci. Nedojde tak k přehlcení této sekce obsahem, který se sice do zařízení stáhl na základě lokací, které uživatel navštívil, ale není pro uživatele zajímavý.

Další částí aplikace je konfigurace. Ta slouží k získání základních studijních informací o uživateli. Aby nebyla konfigurace pro uživatele obtěžující, snažil jsem minimalizovat vyžadované množství informací na minimum. Při prvním spuštění je tedy třeba, aby si uživatel nastavil pouze jedinou věc – aktuální studované obory. Při návrhu bylo nutné počítat i s tím, že jeden student může studovat více oborů a každý v jiném semestru, dále počítat s případem, kdy uživatel není aktuálně studentem. Na obrázku 15 jsou tři obrazovky zajišťující konfiguraci studií. Při prvním otevření mého modulu se zobrazí rovnou obrazovka „Přidej svůj obor“ (obrázek 15 nalevo). Po zvolení studia se zobrazí obrazovka „Vyber semestr“ (obrázek 15 uprostřed). Uvažoval jsem nad dvěma možnostmi – udělat ji jako výběr začátku studia nebo jako výběr aktuálního semestru. Nakonec jsem se rozhodl pro výběr aktuálního semestru, protože je to podle mého názoru údaj, který vyžaduje nižší kognitivní zátěž uživatele, ačkoliv je pak nutné při implementaci tento údaj přepočítávat na

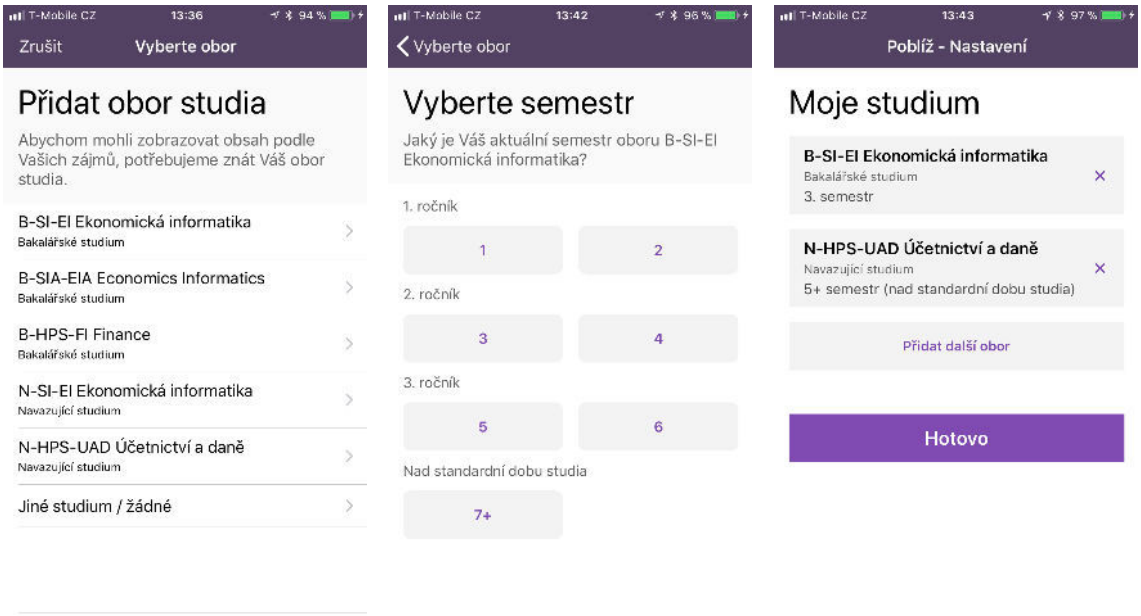
rok začátku studia. Semestry jsem rozdělil podle ročníků pro snadnější orientaci a všechny ročníky přesahující standardní dobu zvoleného studia jsem sloučil do jednoho. Obrazovka s volbou semestru se nezobrazí v případě zvolení „Žádné / jiné studium“. Poslední obrazovkou je „Moje studia“ (obrázek 15 napravo). Na této obrazovce uživatel vidí své aktuálně zvolené studium. Dále může pokračovat přidáním dalšího studia, nebo dokončit konfiguraci pomocí velkého tlačítka „Hotovo“. K této obrazovce se lze kdykoliv vrátit v případě, že si uživatel bude chtít v budoucnu upravit svá studia. Konfigurační aplikace se tak skládá z pouhých tří jednoduchých a přímočarých kroků a neměla by být překážkou pro používání aplikace.

Vzhledem k tomu, že moje aplikace je součástí větší univerzitní aplikace *Smart PEF*, bylo třeba navrhnout i začlenění mého modulu do domovské obrazovky. Název modulu byl zvolen „Poblíž“. Tento název vystihuje základní funkci modulu – zobrazování obsahu vztahujícího se k místům poblíž uživatele. Ikonka modulu je kolečko s orbitem naznačující lokační službu a okolí uživatele. Stejně jako ostatní moduly i modul Poblíž má na úvodní obrazovce rychlé tipy. Pokud ještě uživatel nedokončil úvodní konfiguraci, zobrazí se informace o nutné konfiguraci (obrázek 16 nalevo). V případě dostupného obsahu se zde zobrazí informace, že je dostupný zajímavý obsah (obrázek 16 prostřední). V případě zjištěného problému v nastavení zařízení se zde také zobrazí příslušná informace (např. indikace vypnutého Bluetooth, viz obrázek 16 napravo).

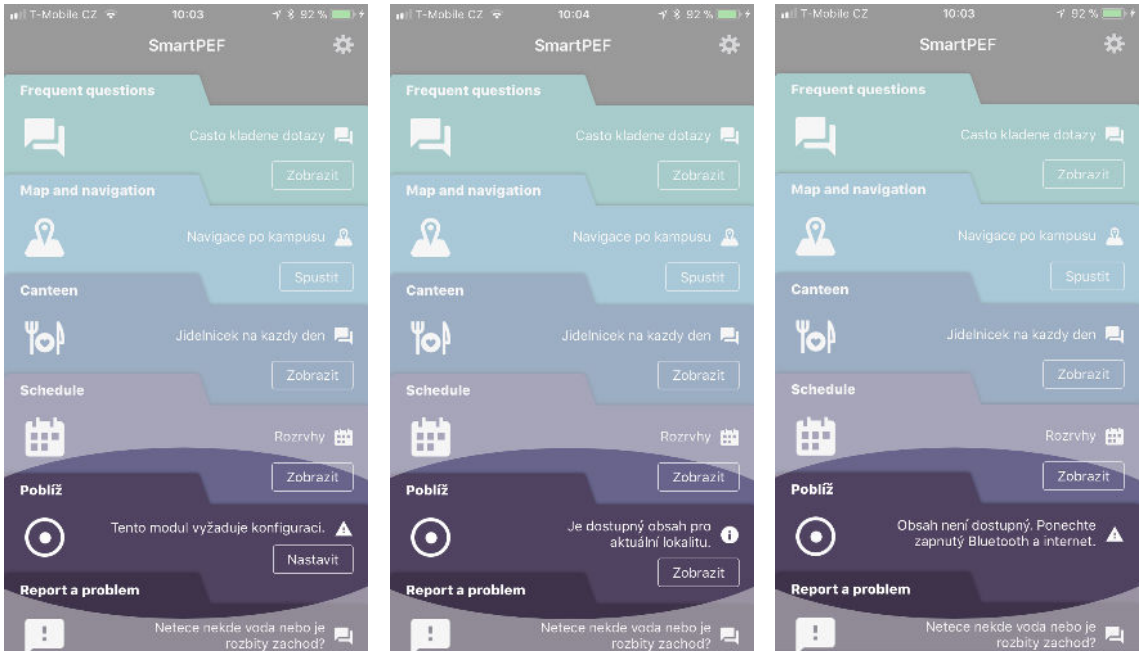
Vzhledem k inovativní povaze této aplikace je potřeba uživatele vhodně zaškolovat do používání systému. Nechtěl jsem vytvářet úvodní obrazovku s návodem, která by spolu s úvodní konfigurací již byla nadměrnou zátěží pro uživatele. Zásadní poučení uživatele jsou proto zobrazeny až později v průběhu používání aplikace. Jedná se např. o text informující o základním principu aplikace – že je potřeba se přiblížit k chytrému místu, aby se v aplikaci zobrazil obsah. Tato informace se zobrazí přímo ve výpisu obsahu v případě, že aktuálně není žádný obsah dostupný (obrázek 17 nalevo). Dalším příkladem poučení uživatele je informace o nutnosti mít stále zapnutý Bluetooth a být připojen k internetu. Toto poučení se zobrazí opět přímo v seznamu obsahu (obrázek 17 ve středu), příp. na hlavní stránce aplikace v menu, jak již bylo popsáno v předchozím odstavci. Podobně jsou také zobrazovány problémy v nastavení notifikací (obrázek 17 napravo) nebo nesprávně nastavená autorizace zjišťování polohy.



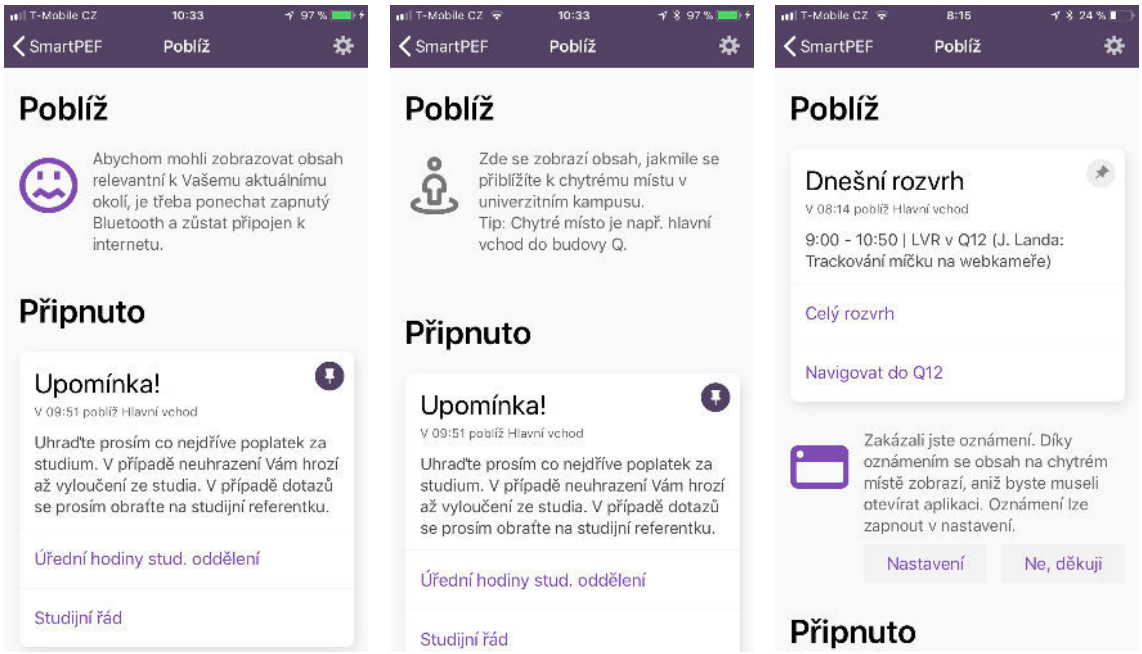
Obrázek 14: Screenshoty z vytvořené mobilní aplikace. Základní obrazovka se seznamem obsahu. První sekce obsahuje obsah související s aktuální lokací uživatele. Druhá sekce obsahuje položky, které si uživatel připnul. Třetí sekce je historie obsahu – obsahuje položky zobrazené uživatelem v posledních 14 dnech.



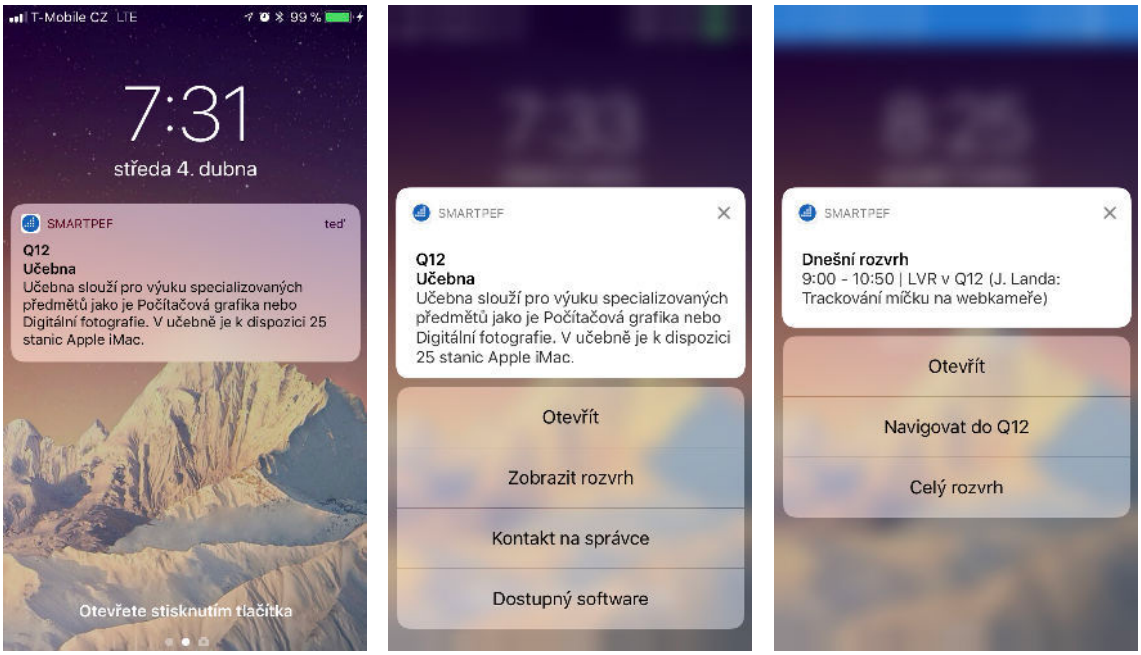
Obrázek 15: Počáteční konfigurace vyžadovaná od uživatele před používáním aplikace. Snažil jsem se o minimalizování počtu kroků, počtu možností a obecně kognitivní zátěže uživatele, aby konfigurace nebyla nepříjemnou překážkou pro používání aplikace.



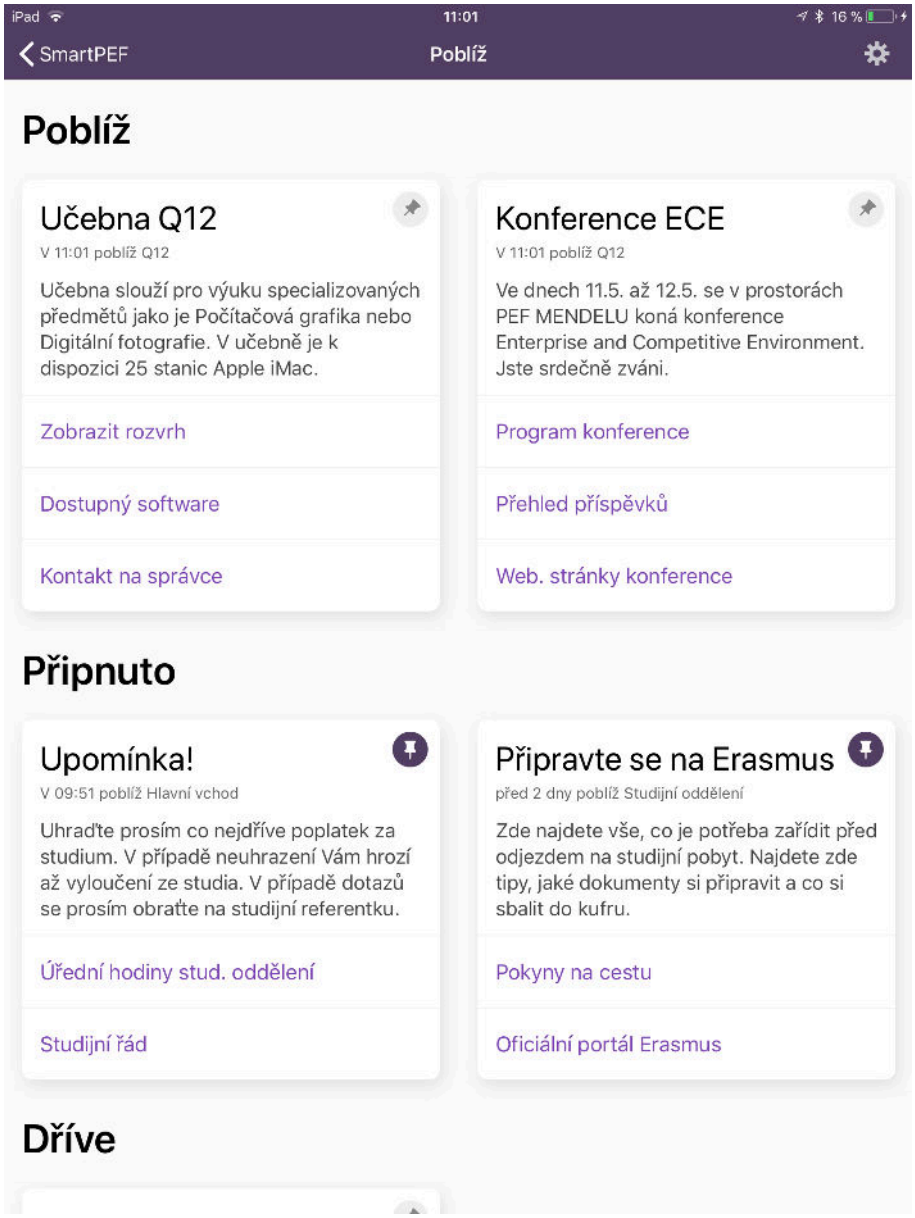
Obrázek 16: Začlenění mé aplikace do hlavní navigace univerzitní aplikace *Smart PEF*. U každé karty napravo se zobrazují rychlé tipy pro daný modul, např. informace o nutné prvotní konfiguraci nebo zjištěné problémy v nastavení telefonu, které jsou překážkou pro správné fungování lokačních služeb.



Obrázek 17: Přímo v seznamu obsahu jsou začleněny poučení uživatele o fungování aplikace nebo problémy s nastavením telefonu. Uživatel je tak poučen o fungování aplikace kognitivně nenáročnou a neobtěžující formou.



Obrázek 18: Ukázky notifikací, které zobrazují obsah podle aktuální lokace. Notifikace jsou bez zvukové a vibrační výstrahy. Tvoří tak pouze rychlý přístup k obsahu a nejsou pro uživatele obtěžující. Navíc je využita funkce *force touch* pro umožnění přímé interakce s obsahem, viz pravé dva obrázky.

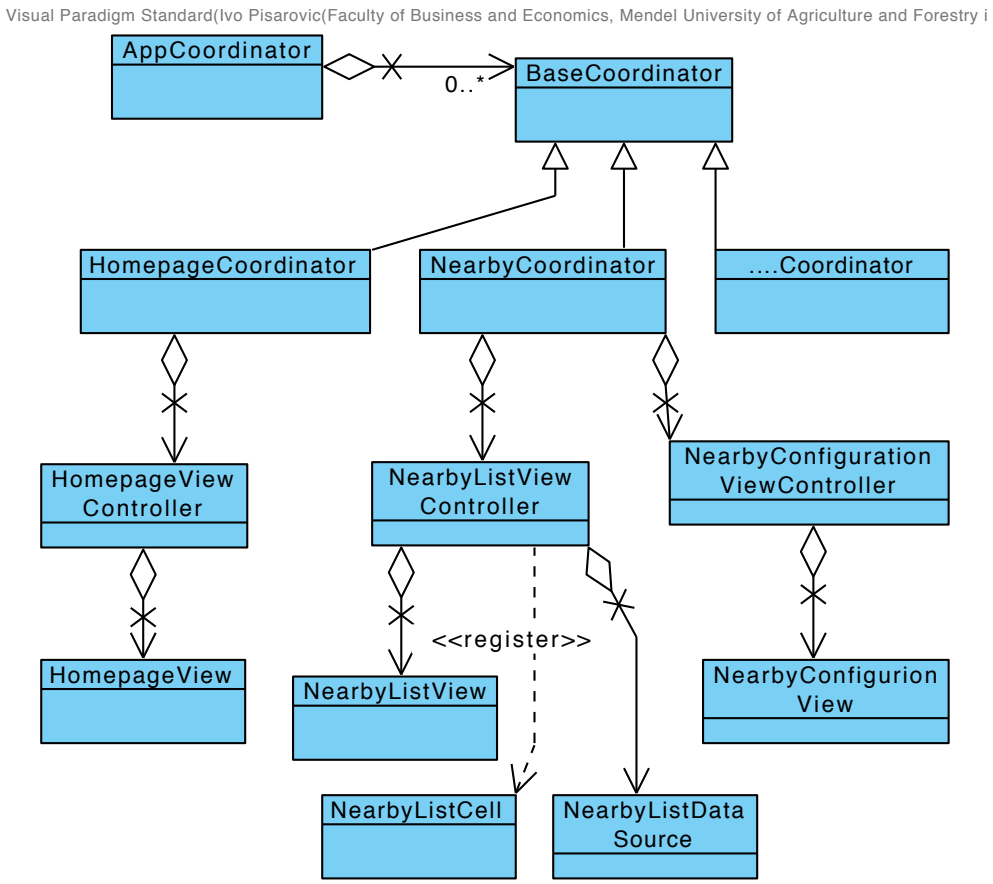


Obrázek 19: Uživatelské rozhraní je připraveno i pro větší displeje. Na obrázku je obsah zobrazený na zařízení *iPad mini 2*.

Architektura

Stejně jako další části univerzitní aplikace *Smart PEF* je i tento modul implementován podle vzoru *Coordinator* (Khanlou, 2015; Townsend, 2016) bez použití storyboardů pro tvorbu uživatelského rozhraní.

Vzor *Coordinator* je značně volný a není jasně vymezen. Jeho cílem je odlehčit třídám typu *ViewController*, které v běžných aplikacích narůstají do obřích rozměrů až několik tisíců řádků kódu. Je to způsobeno především tím, že *ViewController* přestává být objekt a stává se z něj spíše kontejner na metody. Stává se pak, že je jeho obsahem nejen kód řešící stavy uživatelského rozhraní a události navázané na uživatelské rozhraní, ale také v něm je kód zajišťující ukládání dat do databáze nebo stahování dat ze serveru. To však porušuje mnoho základních principů objektového programování a takový objekt již pak není objektem, ale pouze kontejnerem na metody. Podrobněji tento problém se všemi třídami, které končí na *-er*, popisuje Bugayenko (2015). Vzor *Coordinator* převádí část odpovědnosti, zejména základní logiku a přechody mezi obrazovkami, z třídy *ViewController* na třídu *Coordinator*. Díky tomu se snižuje víceúčelovost třídy *ViewController* a lépe se vymezí odpovědnost těchto tříd a návaznost na třídy *View* a *DataSource*. Ukázka, jak byl vzor aplikován v aplikaci *Smart PEF*, je na obrázku 20.



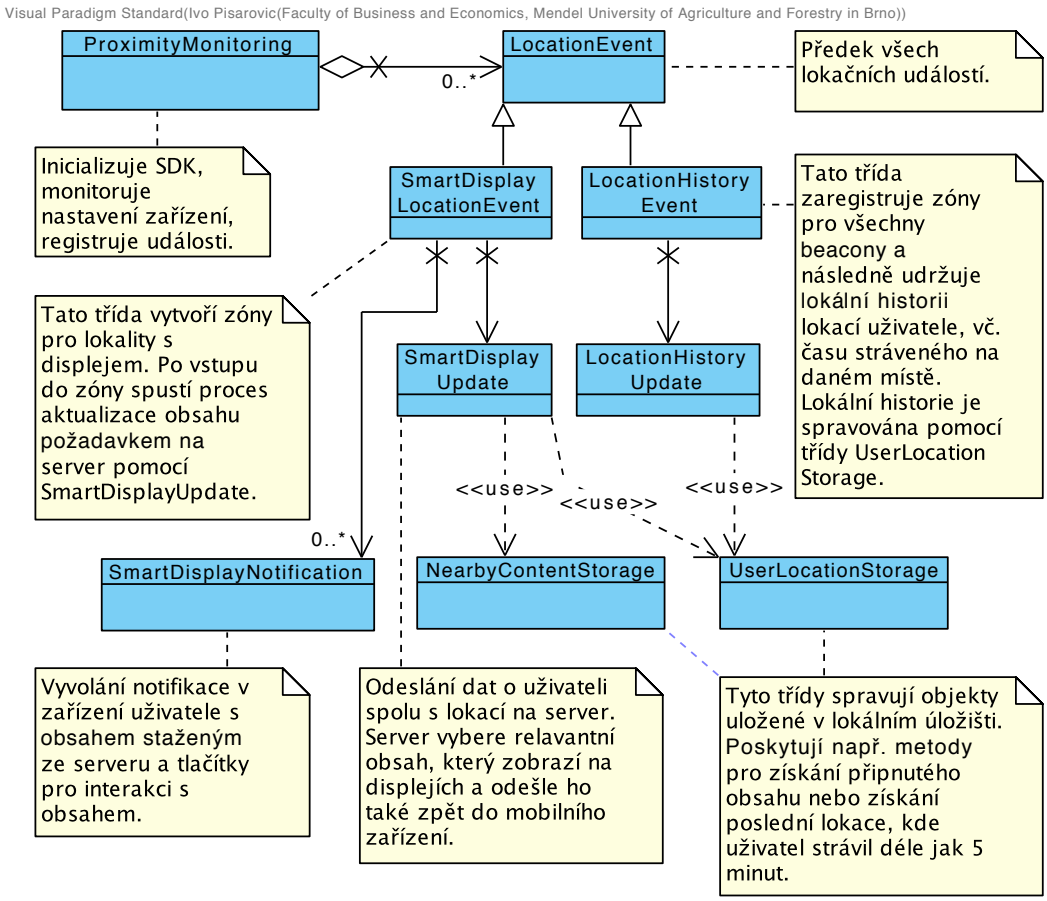
Obrázek 20: Ukázka aplikace návrhového vzoru *Coordinator*.

Uživatelské rozhraní píšeme v rámci aplikace *Smart PEF* přímo ve *Swift* kódu za využití knihovny *SnapKit* (Snapkit, 2017). Tento přístup má řadu výhod zejména pro aplikace většího rozsahu nebo pro aplikace, na kterých spolupracuje mnoho vývojářů. Nevýhody klasického způsobu tvorby uživatelského rozhraní přes *storyboardy* jsou např.: obtížné řešení *merge konfliktů* v nástroji *git*, velmi pomalý nástroj *Interface Builder* pro tvorbu *storyboardů*, absence čitelné kódové reprezentace uživatelského rozhraní, obtížné znovupoužívání vytvořených komponent, nepřehledná tvorba složitějších grafických komponent, nemožnost definovat globální konstanty pro čísla apod.

Aplikace *Smart PEF* je rozdělena na několik modulů: modul uživatelského rozhraní, modul pro správu lokálních dat, synchronizační modul atd. Pro správu závislostí je využit nástroj *CocoaPods*. V rámci mé práce jsem přidal ještě jeden další modul – modul pro události navázané na změnu lokace popsaný dále.

Detekce aktivity pomocí beaconů

V aplikaci *Smart PEF* jsem vytvořil nový modul *LocationEvents*, který řeší lokalizaci pomocí beaconů a vyvolává na změnu lokace navázané akce. Základní struktura tříd této komponenty je na obrázku 21.



Obrázek 21: Struktura komponenty, která řeší události navázané na změnu lokace.

Při testování beaconů jsem však narazil na řadu problémů v souvislosti s použitým *Proximity SDK* od společnosti *Estimote*. Při prvním testování se chování zdálo jako čistě náhodné a nebylo jednoduché chování pochopit a identifikovat chyby v SDK vzhledem k tomu, že nejsou nikde pořádně zdokumentovány.

- Funkce *FlipToSleep* zjednodušuje odladňování mobilní aplikace. Není nutné se fyzicky vzdalovat od beaconu a opětovně k němu přicházet. Místo toho stačí beacon obrátit dnem nahoru a beacon je deaktivován. Bylo však zjištěno, že tato funkce je dobrá spíše na první otestování. Na pořádné testování je nutné se k beaconu přiblížit a vzdálit přirozeně. Je to kvůli tomu, že SDK provádí výpočty pro zlepšení přesnosti detekce založené na postupném přibližování se k beaconu a při pouhém nárazovém otočení není detekce tak přesná a přirozená.
- Nejnovější *Proximity SDK* je závislé na internetu. Internet je potřeba při spuštění aplikace pro stažení dat o beaconech z cloudu. Díky tomu není potřeba nastavovat hodnoty vysílané beacony v terénu a lze je snadno a kdykoliv změnit přes webovou aplikaci od *Estimote* bez nutnosti beacony obcházet s konfigurační aplikací.
- Na platformě iOS můžeme rozlišovat 3 režimy aplikace:
  (1) Aplikace je spuštěna uživatelem a běží na popředí – detekce beaconů je v tomto případě zcela bez problémů.
  (2) Aplikace je spuštěna uživatelem a následně uvedena na pozadí, např. přechodem do menu nebo jiné aplikace – i v tomto případě probíhá detekce bez problémů.
  (3) Aplikace je manuálně vypnuta uživatelem přes přepínač aplikací. Zde nastávají komplikace. Aby systém aplikaci znova spustil na pozadí, musí být zařízení nejméně půl minuty mimo dosah všech beaconů. Jakmile se zařízení opět přiblíží k některému beaconu, aplikace je spuštěna na pozadí. Problémem je, že nestačí být vzdálen od beaconů na vzdálenost nastavenou v aplikaci, ale na maximální dosah beaconu, který může být až desítky metrů. Je to kvůli tomu, že framework *CoreLocation* využívaný pro probuzení aplikace nerozlišuje nastavení dosahu v rámci *Estimote SDK*. Při reálném nasazení to není problém, značně se však komplikuje testování. Tento problém je popsán na fóru *Estimote* (Can't get notification to work when app is killed (Swift), 2017).
- Inicializovat SDK je nutné ihned po spuštění aplikace, např. v metodě `didFinishLaunching()` v třídě `AppDelegate`. V případě probuzení aplikace na pozadí je právě tato metoda zavolána, přičemž v parametru metody je předán důvod oživení.
- Jak už jsem zmínil, je třeba být připojený k internetu v okamžiku inicializace SDK. V případě prvních dvou režimů chodu aplikace popsaných výše jsem to

vyřešil tak, že se monitoruje stav připojení a k inicializaci SDK dojde až při detekci připojení k internetu. V případě režimu 3 však monitoring připojení nemá význam. Poté, co uživatel aplikaci manuálně ukončil a systém aplikaci znova probudil, má aplikace na vyřízení změny lokace několik vteřin. Pokud v této době není zařízení připojeno k internetu, SDK není inicializováno a detekce beaconů nebude probíhat. Detekce může být obnovena až poté, co uživatel aplikaci manuálně spustí. Jinak dojde k opětovnému probuzení, až když uživatel opustí maximální dosah všech beaconů alespoň na půl minuty a znova se přiblíží s připojením k internetu.

- Nové *Proximity SDK* od společnosti Estimote má řadu problémů. Jeden z problémů byl ve verzi 0.11.0, kterou jsem využíval v době testování, protože byla v této době aktuální. V této verzi vůbec nefungovala detekce beaconů na pozadí. Tuto chybu jsem nahlásil v oficiálním repozitáři (Issue #6, 2018). Na základě mého hlášení byla chyba opravena a nová verze SDK byla vydána do pár hodin. Dalším problémem je padání editoru v rámci prostředí Xcode (When EstimoteProximitySDK is imported, the Xcode editor autocomplete crashes, 2018). Padání již bylo později vyřešeno ve verzi 0.11.3. Další problémy pak přišly s novou verzí *Xcode* a verzí jazyka *Swift* 4.1.

I přes výše uvedené nedostatky jsem se rozhodl zůstat u nového *Proximity SDK* z následujících důvodů:

- Původní *Estimote SDK* již bylo označeno jako zastaralé ze strany výrobce a nebude tedy dále podporováno.
- Nové SDK odstraňuje limit 20 oblastí pro monitoring regionů na pozadí stanovený Applem.
- Nové SDK spolehlivě detekuje vstupy a výstupy ze zón. Náhodné události se vyskytují jen zřídka.
- Konfigurace SDK a beaconů je velmi jednoduchá. Stačí beaconům nastavit *key-value* páry přes webovou aplikaci *Estimote Cloud*. V případě detekce beaconu jsou tyto hodnoty předány jako standardní asociativní pole *NSDictionary*. Obsahem mohou být jakékoliv řetězce. Díky tomu zde neplatí omezení dvěma číselnými hodnotami, jak je tomu v případě standardu *iBeacon*.
- Na základě těchto řetězců lze velmi efektivně definovat zóny přímo v aplikaci. Je možné také nadefinovat více zón na jeden beacon, vč. různé detekční vzdálenosti pro různé zóny. (Estimote/iOS-Proximity-SDK, 2018)

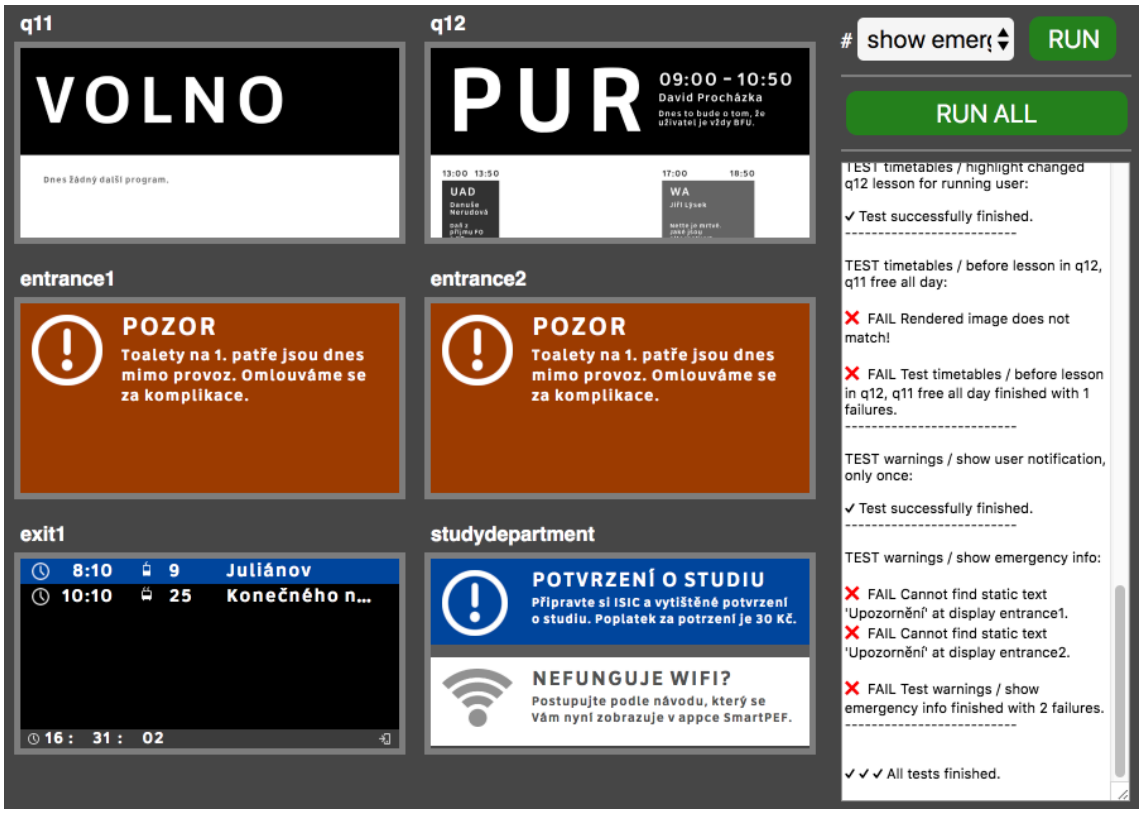
5 Diskuze

5.1 Automatické testování

Pro všechny části systému byly napsány testy, po odladění systému bylo také provedeno expertní testování a uživatelské testování uživatelského rozhraní (popsáno v další sekci).

Mobilní aplikace byla otestována pomocí jednotkových testů a testů uživatelského rozhraní. Aby bylo možné automatizovaně otestovat aplikaci, která je závislá na uživatelském nastavení, na datech ze serveru a na lokaci zařízení, využil jsem tzv. *launch argumenty*. *Launch argumenty* lze nastavit pro každý test. Při spuštění aplikace se pak na základě těchto přepínačů importují testovací data, testovací nastavení, příp. se použije instance serveru určená pro testování.

Serverová aplikace byla otestována jednotkovými a akceptačními testy. Základní testy serveru jsou zajištěny pomocí frameworku *Mocha* (2018). Pro akceptační testování serveru a zobrazovacích aplikací pro displeje jsem vytvořil speciální samostatnou komponentu – webovou aplikaci. V této aplikaci je možné nadefinovat libovolný počet testů. Pro každý test je možné nadefinovat data o uživatelích a prostředí a očekávané výstupy. Při spuštění jsou nejprve tato testovací data předána serveru. Poté se počká, než server aktualizuje displeje. Na závěr je ověřeno, zda se obsah na displejích shoduje s očekávanými výstupy. Vytvořený testovací nástroj podporuje všechny typy displejů – displeje zobrazující obsah ve formě webové aplikace i displeje zobrazující rastrový obrázek. Pro otestování obsahu vyrenderovaného do rastrových obrázků jsem vytvořil webovou aplikaci simulující příslušný displej a umožňující porovnat vyrenderovaný obrázek s očekávaným. V případě objevených problémů mohou být tyto problémy velmi snadno odladěny díky tomu, že testovací nástroj zobrazuje výstup z displejů v reálném čase a je tak možné v případě potřeby přímo vizuálně zkontrolovat zobrazený obsah. Ukázka spuštěných testů je na obrázku 22.



Obrázek 22: Webová aplikace zajišťující akceptační testování. Vlevo je živý náhled na obsah zobrazený na simulovaných displejích. Vpravo jsou ovládací prvky pro spuštění testů a textový výpis s informacemi o proběhlých testech a nalezených problémech.

5.2 Uživatelské testování

Systém byl kromě automatických testů otestován i za účasti uživatelů a bylo také provedeno expertní posouzení.

Testování uživatelského rozhraní se zúčastnilo 5 studentů – dobrovolníků, kteří nebyli nijak zapojeni do vývoje a aplikaci viděli poprvé. Studenti byli stručně uvedeni do systému a byl jim zadán jednoduchý úkol. Jedním z testovacích scénářů bylo reprezentovat studenta přicházejícího ráno do školy a zjistit, kde je následující výuka. Další testovací scénář byl zaměřen na zjištění informací z displeje u učebny zobrazující rozvrh. Poslední scénář testoval navigaci v mobilní aplikaci. Všechny testy dopadly velmi dobře, uživatelé se na displejích i v mobilní aplikaci orientovali bez problémů a bez pomoci zvládli splnit úkoly. Příjemně mě překvapilo, že pro žádného uživatele nebyl problém pochopit význam ikonky v rohu displejů, která indikuje dostupnost obsahu v mobilním zařízení. Při testování bylo zjištěno několik problémů. První problém nastal při zvýraznění výuky červenou barvou, většina uživatelů si myslela, že se jedná o informaci, že jdou pozdě. Tento problém jsem následně odstranil přidáním rámečku k číslu učebny, který naznačuje změnu učebny. Další problém byl s autorizací lokačních služeb – téměř nikdo nečetl popis u žádosti a někteří uživatelé nepovolili během úvodní konfigurace lokační služby a tím pádem byl systém nefunkční. Tento problém jsem částečně vyřešil přidáním varování, které se uživateli zobrazí po dokončení konfigurace v případě, že lokační služby a notifikace nejsou správně nastaveny.

Na základě expertního testování byly odladěny další minoritní problémy. Jednalo se např. o změnu některých pojmů v rámci uživatelského rozhraní, jako např. nahrazení pojmu „výuka“ pojmem „rozvrhová akce“ pro větší obecnost systému, přidání hlavičky s nadpisem na displej zobrazující nejbližší výuku, aby uživatel snáze pochopil smysl displeje, odstranění milisekund ze zobrazeného času apod.

Celkový dojem uživatelů byl velmi pozitivní. Uživatelé snadno získali hledanou informaci z displeje, snadno pracovali s obsahem v mobilní aplikaci a velmi rychle pochopili základní princip fungování systému a způsob propojení jejich lokace, displejů a mobilních zařízení. Uživatelé jednoznačně zhodnotili systém jako přínosný, řešící jejich běžné problémy při pobytu v univerzitním areálu.

5.3 Uživatelská přívětivost

Navržený systém poskytne uživatelům velmi přístupný a pohodlný způsob získávání informací. Systém automatizovaně získává údaje o přítomných uživatelích a jejich chování. Relevantní informace jsou zobrazovány na displejích, které jsou přímou součástí prostředí uživatele podobně jako okna, dveře nebo směrovky na chodbách. Na rozdíl od těchto fyzických objektů je však obsah na displeji dynamický. I přesto zůstávají displeje přirozenou součástí prostředí na rozdíl od mobilních telefonů nebo chytrých hodinek. Zároveň nejsou zatěžující pro běžné každodenní používání, jako je tomu aktuálně v případě brýlí s rozšířenou realitou. Díky těmto vlastnostem systém

poskytuje uživateli informace velmi přívětivým způsobem, bez zvýšené kognitivní zátěže a bez nutnosti provádět manuální akce pro získání informace. Je to podobný přínos jako hledání odjezdu autobusu v mobilní aplikaci ve srovnání s hledáním této informace na zastávkovém jízdním řádu. Pokud je zastávkový jízdní řád ve formě displeje s nejbližšími odjezdy, téměř všichni upřednostní tento displej před mobilním telefonem.

Uživatelská přívětivost navrženého systému značně závisí na obsahu a nastavení jeho parametrů. Velmi negativně by systém mohl působit v případě, kdy by většina obsahu byla spíše reklama, ať už na akce školy nebo na stáže a nabídky firem. Při navrhování obsahu pro můj systém jsem proto kladl důraz na to, aby účel mého systému rozhodně nebyl reklamní. Cílem je usnadnit a zpříjemnit studentům život v areálu univerzity řešením jejich častých problémů. Pokud bude tento hlavní účel dodržen, může být systém vnímán studenty velmi pozitivně.

Na uživatelskou přívětivost má značný vliv také umístění displejů a charakteristiky těchto panelů v návaznosti na konkrétní obsah. Je třeba identifikovat klíčová místa pro umístění displejů, aby mohl být maximálně využit jejich potenciál. Příklady takových míst jsou např.: studijní oddělení nebo prostory před učebnami, kde často čeká více studentů delší dobu, dalším příkladem jsou oblasti vchodů a východů, kde je vysoká intenzita pohybujících se uživatelů. Podle umístění je třeba zvolit vhodný rozměr a typ displeje.

Uživatelské rozhraní mobilní aplikace je velmi jednoduché a přímočaré, obsahuje minimum zanoření, které by znesnadňovaly navigaci. Ukládání obsahu na později je umožněno pomocí přirozeného „připnutí“. A mnohý obsah není třeba ani připínat, protože se automaticky ukládá veškerý obsah zhlédnutý v posledních čtrnácti dnech. Obsah z okolí uživatele, připnutý obsah i historický obsah je zobrazen v jednom seznamu pod sebou a pro jeho procházení stačí skrolovat, což je pro uživatele velmi přirozené. Počáteční konfigurace je snížena na minimum a delší textová poučení uživatele o principu fungování aplikace jsou zobrazena až později přímo v rámci obsahu.

Uživatelské rozhraní displejů obsahuje minimum grafických prvků, využívá kontrastní barvy a speciální písmo, díky čemuž je zajištěna velmi dobrá čitelnost a přehlednost na větší vzdálenost. Díky využití kontextových informací se zobrazuje pouze relevantní obsah. Textového obsahu na displeji je tak co nejméně a lze použít dostatečně velkou velikost písma. Vizuální váha barev a velikosti je využita pro zvýraznění důležitosti.

Propojení displejů a mobilních telefonů značně zjednodušuje uživatelské rozhraní. K aktualizaci obsahu v mobilních zařízeních a na displejích dochází ihned po změně lokace. Pokud není mobilní aplikace aktuálně na popředí, zobrazí se uživateli tichá notifikace s obsahem z okolí. Uživatel si tak může obsah zobrazit nebo provést akci přímo ze zamykací obrazovky, aniž by musel spouštět aplikaci a přecházet do příslušného výpisu položek.

Odezva systému také hraje důležitou roli pro uživatelskou přívětivost. Samotný výběr obsahu a jeho zobrazení je díky použitým technologiím a škálovatelnosti sys-

tému téměř okamžité. Jediným úzkým místem je detekce změny lokace v mobilním zařízení. Doba detekce je u použitých beaconů od společnosti *Estimote* v rozmezí 0–10 vteřin, ve značné části případů (asi polovina) je detekce téměř okamžitá. Záleží však také na umístění zařízení – zda je v ruce nebo v batohu s jinými předměty. Každopádně je nutné s touto prodlevou počítat. V místech, kde je očekávaný větší pohyb uživatelů (např. vstup do budovy), je vhodné umístit více beaconů pro lepší pokrytí a umístit je i do větší vzdálenosti od displeje proti směru chůze uživatelů.

5.4 Bezpečnost

Vzhledem k tomu, že systém pracuje s osobními údaji uživatelů vč. značně citlivých dat, jako je kompletní historie polohy uživatele v univerzitním areálu, bylo třeba stanovit jasná pravidla pro zacházení s daty.

Prvním pravidlem je, že obsah na displeji se nesmí přímo vztahovat ke konkrétnímu uživateli. Pokud musí být zobrazena notifikace pro konkrétního uživatele, musí být formulována obecně bez jména osoby. Detailní informace lze podávat pouze do mobilní aplikace uživatele.

Druhým pravidlem je anonymizace dat. Veškerá data odesílaná z mobilního zařízení na server jsou anonymní. Kvůli personalizování je nutné odesílat na server identifikátor uživatele. Nicméně jedná se pouze o číslo, náhodně vygenerované při prvním spuštění mobilní aplikace, které nelze v rámci serveru přiřadit ke konkrétnímu uživateli.

Třetím pravidlem je uchovávat co nejvíce dat lokálně. Např. historie lokací uživatele je uložena pouze lokálně v zařízení uživatele. Na server se odesílají pouze agregované údaje v minimálním potřebném množství.

Posledním pravidlem je promazávání dat. Na serveru jsou uloženy data uživatelů pouze za poslední půl hodiny. Navíc starší data již ani nejsou potřebná pro výběr relevantního obsahu.

5.5 Obavy uživatelů

Aby systém mohl dobře fungovat, je třeba, aby do něj byla zapojena většina osob z cílové skupiny uživatelů, kteří se pohybují v okolí displejů. Cílovou skupinou jsou myšleni ti, pro které je určen obsah. Např. v současné verzi jde o studenty. Zaměstnanci a vyučující do cílové skupiny nepatří, není pro ně tvořen obsah a tím pádem ani nemusí být do systému zapojeni.

Pro zapojení do systému však čelí uživatelé řadě překážek. V první řadě je třeba navodit u potenciálních uživatelů zvědavost systém vyzkoušet. Zásadním problémem takového systému je fakt, že uživatelé neví, že takový systém potřebují a že ho mohou využít pro svůj prospěch. Poté, co se uživatel rozhodne systém vyzkoušet, je třeba získat od něj základní informace. Překážkou by byla dlouhá a složitá konfigurace s velkým počtem kroků nebo konfigurace nutící uživatele hodně přemýšlet. Taková konfigurace by způsobila, že někteří uživatelé by systém přestaly používat ještě před

dokončením konfigurace. Pokud uživatel dokončí konfiguraci, může již systém naplno využívat.

Aby však systém fungoval neustále, kdykoliv se bude uživatel pohybovat v areálu univerzity, je třeba, aby uživatel byl neustále připojen k internetu a měl zapnutý Bluetooth. Někteří uživatelé tyto služby vypínají, aby šetřili baterii nebo nebyli ničím rušeni. Také je třeba, aby uživatel udělil aplikaci oprávnění pro určování polohy zařízení vč. určování polohy na pozadí. Někteří uživatelé mohou mít obavy o svoji bezpečnost a lokalizaci mohou zakázat.

V budoucnu však lze očekávat snižování těchto bariér, jakmile uživatelé poznají související výhody a začnou těmto službám více důvěřovat. V souvislosti s tímto trendem některá mobilní zařízení již neumožňují permanentně vypnout Bluetooth a připojení k internetu. Díky zvyšování kapacity baterií a nástupu úsporného hardwaru nemá význam vypínání těchto služeb kvůli úspoře baterie. A ani vypínání těchto služeb z důvodu nerušení nemá až takový význam, protože aplikace samy o sobě by měly uživateli dodávat obsah tak, aby respektoval jeho potřeby a nebyl pro něj nikdy rušivý. V ideálním případě by aplikace měla mít informaci o aktuální činnosti uživatele. V případě, že uživatel pracuje, dodávat mu jen urgentní notifikace. Na druhou stranu při dojíždění nebo čekání na chodbě je možné zasílat více notifikací. Ale samozřejmě záleží na preferencích každého uživatele. Preference by bylo možné automaticky identifikovat na základě toho, jestli uživatel při dané činnosti daný typ notifikace jen skryje, nebo zda se přepne do aplikace.

5.6 Využití v praxi

Navržený systém je cílen do prostředí univerzity. Architekturu systému jsem však vytvořil tak, aby jádro systému bylo obecné a nezávislé. Je tedy možné systém libovolně škálovat a využívat i v jiných prostředích než je univerzita.

Systém by jistě našel uplatnění při provozu velkých budov a areálů, jako jsou letiště, nemocnice nebo obchodní domy. V takových prostorách je často obtížné se orientovat, a to zejména v situaci časové tísně. Vytvořený systém může velmi usnadnit navigaci pomocí personalizace směrovek pro konkrétní osoby. Chytré směrovky detekují, že je uživatel příliš daleko od nástupního terminálu, upozorní ho a dovedou na správné místo. Podobný systém může fungovat např. v nemocnici pro sledování a navigaci pacientů.

Dalším příkladem využití jsou velké výrobní podniky. Displeje umístěné ve skladech nebo u výrobních strojů mohou zobrazovat různé informace podle toho, zda daný stroj zrovna obsluhuje dělník, vedoucí výroby, údržbář, nebo manažer podniku. Zatímco dělník ocení informace o probíhajícím obrábění výrobku, vedoucí výroby ocení denní statistiku využití stroje, údržbář stav stroje a manažer výrobní kapacitu, stáří a odhady životnosti.

Zrakově postižené osoby nemohou využívat nástěnných displejů. Na druhou stranu mobilní aplikace poskytuje příhodnou platformu pro sdělování informací handicapovaným osobám. Běžné displeje, tištěné rozvrhy, směrovky a plakáty jsou pro

nevidomé nečitelné. S využitím mého systému by bylo možné informaci z displejů doručit do mobilního zařízení handicapovaného uživatele, kde si ji může poslechnout pomocí převodu textu na řeč. Takové využití by mohlo značně usnadnit handicapovaným navigaci a reakci na okolí v novém prostředí.

6 Závěr

Cílem práce bylo navrhnout a implementovat systém pro cílení obsahu uživatelům na základě jejich lokace, času, informací o uživatelích a okolí.

V kapitole 2 jsem vytvořil literární přehled zaměřený především na lokační služby – jejich využití v reálných aplikacích, lokační služby využitelné pro lokalizaci mobilních zařízení, využití lokačních služeb pro cílení obsahu a jejich využití pro tvorbu uživatelských rozhraní.

V rámci metodiky (kap. 3) jsem navrhl model systému vč. všech jeho komponent a metod pro výběr, přípravu a doručení obsahu do mobilních zařízení a displejů. Přínosem této práce je především propracovaný abstraktní model systému a metoda pro identifikaci uživatelů v okolí displeje.

Kapitola 4 Vlastní práce popisuje implementaci podle navrženého modelu a metod. Implementace zahrnuje server pro poskytování obsahu (sekce 4.1), zobrazovací aplikace pro displeje (sekce 4.2) a mobilní aplikaci pro zobrazování obsahu a sběr dat o uživateli (sekce 4.3). Systém je tvořen mnoha malými, jednoduchými a nezávislými komponentami, které mezi sebou komunikují. Díky tomu ho lze velmi snadno škálovat pro nasazení libovolného rozsahu. Rovněž případná úprava systému pro potřeby jiné domény je velmi snadná. Vzhledem k použitým technologiím a vnitřní jednoduchosti komponent bude budoucí údržba systému jednoduchá i třetí osobou bez většího zaškolování.

V rámci vlastní práce jsem se také zabýval návrhem uživatelského rozhraní. Toto rozhraní je přirozenou součástí prostředí uživatele. Díky automatizovanému využití informací o lokaci uživatele a prostředí je kladena velmi nízká kognitivní zátěž na uživatele. Důležitým prvkem je možnost další interakce s obsahem pomocí mobilních zařízení uživatelů.

V rámci Diskuze (kap. 5) bylo popsáno otestování systému, které probíhalo na několika případech užití cílenými zejména na studenty v univerzitním kampusu. Bylo také provedeno uživatelské a expertní testování. Dále jsem v této kapitole zhodnotil systém z několika hledisek – z hlediska uživatelské přívětivosti a z hlediska bezpečnosti. Na závěr této kapitoly jsem uvedl několik možných využití v rámci univerzity i v jiných organizacích.

Nasazení systému na Provozně ekonomické fakultě Mendelovy univerzity v Brně je plánováno do konce roku společně s dalšími moduly v rámci univerzitní aplikace *Smart PEF*.

7 Reference

About privacy and Location Services in iOS 8 and later. *Apple Support* [online]. [cit. 2017-10-14]. Dostupné z: https://support.apple.com/en-us/HT203033.

Accent Systems [online]. [cit. 2017-10-16]. https://accent-systems.com/.

Adams, M. Roadgeek 2005 font. *FontSpace* [online]. [cit. 2018-01-10]. Dostupné z: http://www.fontspace.com/michael-d-adams/roadgeek-2005.

Adomavicius G., Tuzhilin A. Context-Aware Recommender Systems. *Ricci F., Rokach L., Shapira B. (eds) Recommender Systems Handbook.* Springer, Boston, MA. DOI: 10.1007/978-1-4899-7637-6_6.

Affinity - Professional creative software. *Serif Ltd.* [online]. [cit. 2017-10-17]. Dostupné z: https://affinity.serif.com/en-gb/.

Al-Rahab, M. M., Alkheder, S. A., Hoshang, S. A. An Intelligent Location-Based Service System (ILBSS) using mobile and spatial technology: A proposal for Abu Dhabi petrol stations. *Case Studies on Transport Policy* [online]. 2017, 5(2), 245-253 [cit. 2017-10-04]. DOI: 10.1016/j.cstp.2017.01.005. ISSN 2213624X.

Alexandru, B. Over 100 use cases and examples for iBeacon technology. *MoWOW studios* [online]. [cit. 2017-10-04]. Dostupné z: http://blog.mowowstudios.com/2015/02/100-use-cases-examples-ibeacon-technology/.

Ali, K., Javed, T., Hassanein, H. S., Oteafy, S. M. A. Non-audible acoustic communication and its application in indoor location-based services. *2016 IEEE Wireless Communications and Networking Conference* [online]. IEEE, 2016, s. 1-6 [cit. 2017-10-14]. DOI: 10.1109/WCNC.2016.7565076. ISBN 978-1-4673-9814-5.

Alt, F., Shirazi, A. S., Kubitza, T., Schmidt, A. Interaction techniques for creating and exchanging content with public displays. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13* [online]. New York, USA: ACM Press, 2013, s. 1709- [cit. 2017-10-17]. DOI: 10.1145/2470654.2466226. ISBN 9781450318990.

Anderson, J. The icon of modern art puts Estimote beacons on display. *Reality matters. The Estimote Team Blog* [online]. 2017-02-20 [cit. 2018-02-15]. Dostupné z: http://blog.estimote.com/post/157200820650/the-icon-of-modern-art-puts-estimote-beacons-on.

App Store - Support. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/support/app-store/.

App Store Review Guidelines. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/app-store/review/guidelines/#location.

Background Execution. *Apple Developer Documentation* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/library/content/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/BackgroundExecution/BackgroundExecution.html#//apple_ref/doc/uid/TP40007072-CH4-SW1.

BBLocationManager. *GitHub* [online]. [cit. 2017-10-14]. Dostupné z: https://github.com/benzamin/BBLocationManager.

Beacon FAQ. *Locatify* [online]. [cit. 2017-10-14]. Dostupné z: https://locatify.com/blog/beacon-faq/.

Beacons. *Google Developers* [online]. [cit. 2017-10-14]. Dostupné z: https://developers.google.com/beacons/.

Blue Sense Networks [online]. [cit. 2017-10-16]. http://bluesensenetworks.com/.

BlueCats Beacons [online]. [cit. 2017-10-16]. https://www.bluecats.com/.

Broussard, M. Twitter Begins iOS Rollout of Location-Based Tweet Aggregating Feature. *Mac Rumors* [online]. [cit. 2017-10-04]. Dostupné z: https://www.macrumors.com/2016/06/24/twitter-location-based-tweet/.

Buckler, C. SQL vs NoSQL: The Differences. *SitePoint* [online]. 2015-08-18 [cit. 2018-02-16]. Dostupné z: https://www.sitepoint.com/sql-vs-nosql-differences/.

Buckler, C. SQL vs NoSQL: How to Choose. *SitePoint* [online]. 2015-09-23 [cit. 2018-02-16]. Dostupné z: https://www.sitepoint.com/sql-vs-nosql-choose/.

Bugayenko, Y. Don't Create Objects That End With -ER. *Yegor256.com.* [online]. 2015-03-09. [cit. 2018-04-04]. Dostupné z: http://www.yegor256.com/2015/03/09/objects-end-with-er.html.

Build your own Mirror template. *Developer.estimote.com* [online]. [cit. 2017-10-17]. Dostupné z: http://developer.estimote.com/mirror/build-your-own-template/.

Can't get notification to work when app is killed (Swift). *Estimote Community Forums.* [online]. 2017-10-27. [cit. 2018-04-04]. Dostupné z: https://forums.estimote.com/t/cant-get-notification-to-work-when-app-is-killed-swift/7485.

Cast content to Mirror from a mobile app. *Developer.estimote.com* [online]. [cit. 2017-10-17]. Dostupné z: http://developer.estimote.com/mirror/cast-from-a-mobile-app/.

Choi, E., Song, H., Lee, J., Bae, Ch. S. ACAM: Personalized application managements based on application usage and location. *2013 International Conference on ICT Convergence (ICTC)* [online]. IEEE, 2013, s. 988-991 [cit. 2017-10-04]. DOI: 10.1109/ICTC.2013.6675537. ISBN 978-1-4799-0698-7. Dostupné z: http://ieeexplore.ieee.org/document/6675537/.

Context awareness. *Wikipedia, The Free Encyclopedia* [online]. [cit. 2017-10-05]. Dostupné z: https://en.wikipedia.org/wiki/Context_awareness.

Cooper, A. About face: the essentials of interaction design. 4th edition. Indianapolis, John Wiley, 2014. ISBN 978-1-118-76657-6..

Core Location. *Apple Developer Documentation* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/documentation/corelocation.

Danova, T. BEACONS: What They Are, How They Work, And Why Apple's iBeacon Technology Is Ahead Of The Pack. *Business Insider* [online]. [cit. 2017-10-16]. http://www.businessinsider.com/beacons-and-ibeacons-create-a-new-market-2013-12.

Danova, T. The Hitchhikers Guide to iBeacon Hardware: A Comprehensive Report by Aislelabs. *Aislelabs* [online]. [cit. 2017-10-16]. https://www.aislelabs.com/reports/beacon-guide/.

Duque, J. FC Barcelona builds smart spaces in Camp Nou with beacons. *Reality matters. The Estimote Team Blog* [online]. 2015-11-06 [cit. 2018-02-15]. Dostupné z: http://blog.estimote.com/post/132670826675/fc-barcelona-builds-smart-spaces-in-camp-nou-with.

Estimote, Inc. [online]. [cit. 2017-10-04]. Dostupné z: https://estimote.com/.

Estimote/iOS-Proximity-SDK. *GitHub.* [online]. [cit. 2018-04-04]. Dostupné z: https://github.com/Estimote/iOS-Proximity-SDK/#example.

Express. Node.js web application framework. [online]. [cit. 2018-04-04]. Dostupné z: https://expressjs.com/.

Farias, L. Why is Node.js Becoming the Top Programming Language in Enterprise?. *Concepta* [online]. [cit. 2018-02-16]. https://conceptainc.com/blog/why-is-node-js-becoming-top-programming-language-in-enterprise/.

Feng, J., Liu, Y. Intelligent Context-Aware and Adaptive Interface for Mobile LBS. *COMPUTATIONAL INTELLIGENCE AND NEUROSCIENCE.* 2015. ISSN: 16875265.

Fonts that can be read at a distance. *Ask.metafilter.com* [online]. 2011-11-08 [cit. 2018-01-10]. Dostupné z: https://ask.metafilter.com/200413/Fonts-that-can-be-read-at-a-distance.

Fridman, L., Weber, S., Greenstadt, R., Kam, M. Active Authentication on Mobile Devices via Stylometry, Application Usage, Web Browsing, and GPS Location. *IEEE Systems Journal* [online]. 2017, 11(2), 513-521 [cit. 2017-10-13]. DOI: 10.1109/JSYST.2015.2472579. ISSN 1932-8184.

GCell iBeacon [online]. [cit. 2017-10-16]. https://ibeacon.solar.

Getting Started with iBeacon. Version 1.0. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf.

Getting the User's Location. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/LocationAwarenessPG/CoreLocation/CoreLocation.html.

Gimbal [online]. [cit. 2017-10-16]. Dostupné z: https://gimbal.com/.

Girish, D. 4 Mistakes to Avoid When Developing Your Beacon App. *Beaconstac* [online]. [cit. 2017-10-15]. Dostupné z: https://blog.beaconstac.com/2016/02/4-mistakes-to-avoid-when-developing-your-beacon-app/.

Glimworm Beacons [online]. [cit. 2017-10-16]. https://glimwormbeacons.com/.

Google Play Google Inc. [online]. [cit. 2017-07-28]. Dostupné z: https://play.google.com/store.

Hamad International Airport launches mobile app with iBeacon features. *PASSENGER SELF SERVICE* [online]. 2016-03-08 [cit. 2018-02-15]. Dostupné z: http://www.passengerselfservice.com/2016/03/hamad/.

Hoober, S., Berkman, E. Designing mobile interfaces. Sebastopol, CA: O'Reilly, 2012. ISBN 1449394639.

Hörold, S., Mayas, C., Krömker, H. Interactive Displays in Public Transport – Challenges and Expectations. *Procedia Manufacturing* [online]. 2015, 3, 2808-2815 [cit. 2017-10-13]. DOI: 10.1016/j.promfg.2015.07.932. ISSN 23519789.

iBeacon. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/ibeacon/.

IngeoSDK - Power efficient location framework for iOS [online]. [cit. 2017-10-14]. Dostupné z: http://ingeo.io/sdk/.

IOS 10. *En.wikipedia.org* [online]. [cit. 2017-10-16]. https://en.wikipedia.org/wiki/IOS_10.

Issue #6. 0.11.0 not working in background. Estimote/iOS-Proximity-SDK. *GitHub.* [online]. [cit. 2018-04-04]. Dostupné z: https://github.com/

Estimote/iOS-Proximity-SDK/issues/6.

iTunes Apple Inc. [online]. [cit. 2017-07-28]. Dostupné z: https://itunes.apple.com.

Kazankov, V. PHP VS NODE.JS. *Belitsoft* [online]. 2017-10-12 [cit. 2018-02-16]. Dostupné z: https://belitsoft.com/php-development-services/php7-vs-nodejs/.

Khanlou, S. Coordinators Redux. *Khanlou.com* [online]. 2015-10-05. [cit. 2018-04-04]. Dostupné z: http://khanlou.com/2015/10/coordinators-redux/.

Kontakt.io [online]. [cit. 2017-10-16]. https://kontakt.io/.

Křížová, T.IBeacon – nová technologie v obchodě a cestovním ruchu. České Budějovice, 2017. Bakalářská práce. Jihočeská univerzita v Českých Budějovicích..

Li, Y., Thuang, Y., Zhang, P., Lan, H., Niu, X., El-Sheimy, N. An improved inertial/wifi/magnetic fusion structure for indoor navigation. *Information Fusion* [online]. 2017, 34, 101-119 [cit. 2017-10-17]. DOI: 10.1016/j.inffus.2016.06.004. ISSN 15662535.

Liu, H, Wu, G., Wang, G. Tell me where to go and what to do next, but do not bother me. *Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14* [online]. [cit. 2017-10-05]. DOI: 10.1145/2645710.2645718. ISBN 9781450326681. Dostupné z: http://dl.acm.org/citation.cfm?doid=2645710.2645718.

Location awareness. *Wikipedia, The Free Encyclopedia* [online]. [cit. 2017-10-05]. Dostupné z: https://en.wikipedia.org/wiki/Location_awareness.

LokiJS: Lightweight javascript in-memory database [online]. [cit. 2018-02-16]. http://lokijs.org/.

Mallik, N. 5 Best iBeacon apps that are leading the pack. *Beaconstac* [online]. [cit. 2017-10-04]. Dostupné z: https://blog.beaconstac.com/2015/02/5-best-ibeacon-apps-that-are-leading-the-pack/.

Martin, E., Vinyals, O., Friedland, G., Bajcsy, R. Precise indoor localization using smart phones. *Proceedings of the international conference on Multimedia - MM '10* [online]. New York, USA: ACM Press, 2010, s. 787- [cit. 2017-10-14]. DOI: 10.1145/1873951.1874078. ISBN 9781605589336.

Mittal, S. 4 Interesting iBeacon Use cases You Need to Know. *Beaconstac* [online]. [cit. 2017-10-04]. Dostupné z: https://blog.beaconstac.com/2016/06/4-interesting-ibeacon-use-cases-you-need-to-know/.

Mocha. The fun, simple, flexible JavaScript test framework. *Mochajs.org.* [online]. [cit. 2018-04-04]. Dostupné z: https://mochajs.org/.

Mortensen, D. Natural User Interfaces – What are they and how do you design user interfaces that feel natural? *The Interaction Design Foundation* [online]. [cit. 2017-10-05]. Dostupné z: https://www.interaction-design.org/literature/article/natural-user-interfaces-what-are-they-and-how-do-you-design-user-interfaces-that-feel-natural.

Nearby. *Google Developers* [online]. [cit. 2018-02-15]. Dostupné z: https://developers.google.com/nearby/.

Nette – jak vytvořit REST API – json. *vEnCa-X blog* [online]. [cit. 2017-10-17]. Dostupné z: https://blog.venca-x.cz/nette-jak-vytvorit-rest-api-json/.

Nielsen, J. 10 Usability Heuristics for User Interface Design. *Nielsen Norman Group* [online]. [cit. 2017-10-05]. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/.

node-gd. npm [online]. [cit. 2018-02-16]. Dostupné z: https://www.npmjs.com/package/node-gd.

Otwell, T. HTTP Controllers - Laravel - The PHP Framework For Web Artisans. *Laravel.com* [online]. [cit. 2017-10-17]. Dostupné z: https://laravel.com/docs/5.1/controllers.

Perfitt, T. 10 Things You Need to Know About iBeacon Deployments. *LinkedIn Pulse* [online]. [cit. 2017-10-15]. Dostupné z: https://www.linkedin.com/pulse/10-things-you-need-know-ibeacon-deployments-timothy-perfitt.

PHP Image Workshop - PHP class using GD library for image processing [online]. [cit. 2017-10-17]. Dostupné z: http://phpimageworkshop.com/.

PHP: GD - Manual. *Php.net* [online]. [cit. 2017-10-17]. Dostupné z: http://php.net/manual/en/book.image.php.

PhpStorm: Lightning-Smart IDE for PHP Programming by JetBrains. *JetBrains* [online]. [cit. 2017-10-17]. Dostupné z: https://www.jetbrains.com/phpstorm/.

Procházka, D. *Studijní materiály k předmětu Pokročilá uživatelská rozhraní.* Mendelova univerzita v Brně, 2017.

Qt | Cross-platform software development for embedded & desktop. *Qt.io* [online]. [cit. 2017-10-17]. https://www.qt.io/.

Radius Networks [online]. [cit. 2017-10-16]. https://www.radiusnetworks.com/.

Rahaman, H., Biswas, A. B., Nazimuddin, S. M., Rahman, E., Khan, R. Synchronous location-aware media and augmented visualization for real world tourist (SMART): An application for Khalifatabad heritage site, Bagerhat, Bangladesh. In: *2016 22nd International Conference on Virtual System & Multimedia (VSMM)* [online]. IEEE, 2016, s. 1-7 [cit. 2017-10-04]. DOI:

10.1109/VSMM.2016.7863159. ISBN 978-1-4673-8993-8. Dostupné z: http://ieeexplore.ieee.org/document/7863159/.

Raspberry Pi - Teach, Learn, and Make with Raspberry Pi [online]. [cit. 2017-10-17]. Dostupné z: https://www.raspberrypi.org/.

Region Monitoring and iBeacon. *Apple Developer* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/LocationAwarenessPG/RegionMonitoring/RegionMonitoring.html.

Remote Display | Cast. *Google Developers* [online]. [cit. 2017-10-17]. Dostupné z: https://developers.google.com/cast/docs/remote.

Sanchez, R. Comparing Node.js vs PHP Performance. *HostingAdvice.com* [online]. 2018-01-11 [cit. 2018-02-16]. Dostupné z: http://www.hostingadvice.com/blog/comparing-node-js-vs-php-performance/.

Set-simulator-location. *GitHub* [online]. [cit. 2017-10-14]. Dostupné z: https://github.com/lyft/set-simulator-location.

Snapkit [online]. [cit. 2017-10-17]. Dostupné z: http://snapkit.io/.

SOMotionDetector. *SocialObjects-Software, GitHub* [online]. [cit. 2017-10-14]. Dostupné z: https://github.com/SocialObjects-Software/SOMotionDetector.

Son, S., Shin, Y. Design of Smart Shopping Application Using Barcode Scanning and Location Based Coupon Service. In: *2015 8th International Conference on Grid and Distributed Computing (GDC)* [online]. DOI: 10.1109/GDC.2015.18. ISBN 978-1-4673-9846-6. Dostupné z: http://ieeexplore.ieee.org/document/7433749/.

Stawarz, K., Cox, A. L., Blandford, A. Beyond Self-Tracking and Reminders. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* [online]. New York, New York, USA: ACM Press, 2015, s. 2653-2662 [cit. 2017-10-13]. DOI: 10.1145/2702123.2702230. ISBN 9781450331456.

Swift. *Apple Developer* [online]. [cit. 2017-10-17]. Dostupné z: https://developer.apple.com/swift/.

SwiftLocation. *GitHub* [online]. [cit. 2017-10-14]. Dostupné z: https://github.com/malcommac/SwiftLocation/.

Tesco Lotus [online]. [cit. 2018-02-15]. Dostupné z: https://www.tescolotus.com/en/home.

The digital design toolkit. *Sketch* [online]. [cit. 2017-10-17]. Dostupné z: https://www.sketchapp.com/.

Tidwell, J. Designing interfaces. 2nd ed. Sebastopol, CA: *O'Reilly*, 2011. ISBN 9781449379704..

Tomitsch, M., Ackad, Ch., Dawson, O., Hespanhol, L., Kay, J. Who cares about the Content? An Analysis of Playful Behaviour at a Public Display. *Proceedings of The International Symposium on Pervasive Displays - PerDis '14* [online]. New York, New York, USA: ACM Press, 2014, s. 160-165 [cit. 2017-10-13]. DOI: 10.1145/2611009.2611016. ISBN 9781450329521.

Top 12 Best PHP RESTful Micro Frameworks (Pro/Con). *Gajotres.net* [online]. [cit. 2017-10-17]. Dostupné z: https://www.gajotres.net/best-available-php-restful-micro-frameworks/.

Townsend, W. An iOS Coordinator Pattern. *Will.Townsend.io.* [online]. 2016-12-18. [cit. 2018-04-04]. Dostupné z: https://will.townsend.io/2016/an-ios-coordinator-pattern.

tvOS for Developers. *Apple Developer* [online]. [cit. 2017-10-17]. Dostupné z: https://developer.apple.com/tvos/.

Using the Significant-Change Location Service. *Apple Developer Documentation* [online]. [cit. 2017-10-14]. Dostupné z: https://developer.apple.com/documentation/corelocation/getting_the_user_s_location/using_the_significant_change_location_service.

Vendors & Manufacturers of iBeacon or Bluetooth BLE Beacons. *Qliktag.com* [online]. [cit. 2017-10-15]. Dostupné z: https://www.qliktag.com/list-of-top-vendors-manufacturers-of-ibeacon-or-bluetooth-ble-beacons/.

Vossen, G., Dillon, S., Schomm, F., Stahl, F. A Classification Framework for Beacon Applications. *Open Journal of Internet of Things (OJIOT)* [online]. Research online Publishing, 2017, 3(1). [cit. 2018-01-04]. ISSN 2364-7108.

What is a beacon protocol? Can beacons broadcast multiple packets simultaneously? *Estimote Community Portal* [online]. [cit. 2017-10-22]. Dostupné z: https://community.estimote.com/hc/en-us/articles/208546097-What-is-a-beacon-protocol-Can-beacons-broadcast-multiple-packets-simultaneously-.

When EstimoteProximitySDK is imported, the Xcode editor autocomplete crashes. Issue #5. Estimote/iOS-Proximity-SDK. *GitHub.* [online]. [cit. 2018-04-04]. Dostupné z: https://github.com/Estimote/iOS-Proximity-SDK/issues/5.

Xcode. *Apple Developer* [online]. [cit. 2017-10-17]. Dostupné z: https://developer.apple.com/xcode/.

Zafari, F., Papapanagiotou, I., Devetsikiotis, M. and Hacker, T. An iBeacon based Proximity and Indoor Localization System. *Arxiv.org* [online]. [cit. 2017-10-14]. Dostupné z: https://arxiv.org/abs/1703.07876.

Přílohy

A Elektronické přílohy

V elektronické podobě jsou k této práci přiloženy následující soubory:

- návrh uživatelského rozhraní mobilní aplikace a displejů vč. zdrojových souborů ve formátu pro nástroj *Sketch*,
- zdrojový kód mobilní aplikace (samostatně nefunkční, součást většího projektu *Smart PEF*),
- zdrojový kód serverové služby obsahující:
  - serverovou aplikaci pro výběr a poskytování obsahu,
  - aplikaci pro displeje sloužící k zobrazování obsahu,
  - několik připravených šablon pro různé typy obsahu,
  - ukázková data připravená jako import do databáze,
  - návod na první spuštění serverové služby a nastavení displejů,
  - testovací nástroj pro akceptační testování,
  - sadu testů pokrývající všechny navržené případy užití,
  - popis REST rozhraní serveru,
  - popis struktury databáze.