Sem vložte zadání Vaší práce.

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

Bakalářská práce

SAGELab - REST rozhraní zařízení SAGE

Ondřej Brém

Vedoucí práce: Ing. Jiří Melnikov

7. května 2015

Poděkování

Na tomto místě bych chtěl poděkovat svému vedoucímu Jiřímu Melnikovovi za cenné rady a pomoc při realizaci této práce. Dále bych chtěl poděkovat členům týmu podílejícím se na projektu SAGElab za plodné dikuze a vzájemnou podporu. Jmenovitě se jendá o kolegy Data Pham Tat, Patrika Faistavera a Jiřího Kubištu. Rád bych také poděkoval svým kamarádům Martinovi a Kristýně Ansorge za jazykovou korekturu práce. V neposlední řadě děkuji svým rodičům za to, že mi umožnili studovat vysokou školu.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 7. května 2015 .....

Odkaz na tuto práci

Brém, Ondřej. *SAGELab - REST rozhraní zařízení SAGE*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato práce se zaměřuje na zpřístupnění technologií Ultra HD v laboratoři SAGE široké veřejnosti. Laboratoř umožňuje přístup k telestěně uživatelům z širokého spektra oborů mimo IT. Tato práce má umožnit propojení uživatelsky přívětivých UI zařízení s telestěnou tak, aby byla odstraněna technologická bariéra v jejím využití. Rešeršní část práce analyzuje možnosti systémy SAGE2, který slouží k obsluze telestěny a možnosti napojení různých ovládacích prvků na tento systém. V další části práce je navrženo a implementováno API pro propojení telestěny s aplikací na mobilní platformě Android. Rozhraní je navrženo tak, aby jeho využití bylo co nejsnazší a nejuniverzálnější. Zároveň je kladen důraz na poskytnutí funkcí nezbytných k dobrému UI na platformě Android. V závěru práce jsou shrnuty výsledky testování funkčnosti rozhraní a jeho propojení jak s telestěnou, tak s aplikací platformy Android.

Klíčová slova SAGE2, UHD, Javascript, REST, User Interface, NodeJS, Android

Abstract

This work deals with making the Ultra High Definition technology in SAGE laboratory accessible to wide public. The lab allows access to the telewall to

users from wide range of fields of expertise outside the IT. This work's goal is to allow connecting user friendly UI devices with the telewall, so that the technological barrier to use it is gone. The research part analysis the available options of SAGE2 software, which is used to run the wall and the options of connecting different UI devices to the system. In the next part the API for connecting the Android app and the wall is designed and implemented. The API is designed so that it is universal and easy to use. At the same time, providing the right functions for the Android app to be nice and easy to use, is crucial. The results of testing the API itself and the connections with Android app and the wall are covered at the end of the paper.

Keywords SAGE2, UHD, Javascript, REST, User Interface, NodeJS, Android

Obsah

Seznam obrázků

Úvod

Každá návštěva obchodu či e-shopu s elektronikou nás upozorňuje na HD či UltraHD rozlišení a zařízení, která jím nedisponují, jako by ani nebyla. Jak to ale ve skutečnosti je s využitelností a praktickou použitelností těchto úžasných novinek? Na to se snaží dát alespoň částečnou odpověď SAGElab[1].

SAGElab je společným pracovištěm sdružení CESNET, Fakulty informačních technologií ČVUT a Fakulty elektrotechnické ČVUT. S vysokým rozlišením se zde setkáte opravdu na každém kroku. Důkazem jsou experimenty na 10K telestěně[2].

Jak napovídá název práce, nebudeme se primárně bavit o rozlišení přístrojů, ale o přiblížení laboratoře běžným uživatelům. V rámci týmové spolupráce zde probíhá vývoj ovládání pomocí rozličných rozhraní, ať už se jedná o zařízení Kinect, nebo o mobilní tabletovou aplikaci či webové rozhraní.

V rámci této práce se čtenář v úvodní kapitole seznámí s laboratoří SAGE a projektem na její rozvoj. V kapitole analýza bude navázáno na úvod do projektu SAGE bližším seznámením se současným stavem SW vybavení laboratoře. Na základě analýzy případů využití laboratoře a jejího vybavení budou v další části analýzy sestaveny funkční a nefunkční požadavky na navrhované API. Toto API bude sloužit k napojení mobilních UI aplikací na SAGE2. V závěru analýzy bude provedeno zhodnocení možností realizace požadavků s ohledem na poskytovanou funkčnost ze strany SAGE2 a dalších doplňkových aplikací.

V návrhové části práce bude popsán obecný princip realizace a jak bude aplikace propojena s ostatními součástmi projektu SAGE.

Na základě návrhu bude v realizační části práce popsána implementace a poskytované API. Zároveň budou zhodnoceny použité technologie a možnosti systému SAGE2 pro realizaci stanovených funkčních požadavků.

[1] Síťová multimediální laboratoř na FIT ČVUT

[2] Telestěna je v rámci této práce chápána jako stěna složená z několika (20) monitorů spojených tak, aby vystupovaly jako jeden monitor.

Nakonec bude ve spolupráci s ostatními autory propojovaných aplikací provedeno testování funkčnosti a použitelnosti výsledků projektu SAGE.

Rozbor zadání

V rámci práce bude provedena analýza softwaru SAGE2 se zaměřením na API jím poskytované. Dále budou zjištěny možnosti napojení ovládání hardwaru (HW) telestěny do jednotného API.

Podle možností poskytnutých SAGE2 a HW telestěny bude navrženo vhodné API mezivrstvy tak, aby vyhovovalo požadavkům pro využití na mobilních platformách. Zhodnocení požadavků mobilních platforem bude provedeno na základě práce zaměřené na vývoj UI aplikace pro OS Android.

Podpora autentizace bude řešena formou volání připraveného API ověřovacího systému shibboleth. V případě nedostupnosti tohoto API bude řešení pouze připraveno na doplnění této funkčnosti.

Po nasazení v SAGElabu bude aplikace testována pomocí uživatelských i automatických testů. Dále bude proveden test propojení s aplikací na OS Android.

Cíle práce

- seznámit se se softwarem SAGE2 a jeho technickými možnostmi
- seznámit se s technologií nodeJS a REST
- navrhnout rozhraní REST pro ovládání SAGE2 pomocí mobilních aplikací
- realizovat server poskytující toto rozhraní
- seznámit se s technologií shibboleth a využít ji k ověření uživatele

SAGElab

Cíle a využití

SAGElab, jinak řečeno Síťová a multimediální laboratoř, byla založena a je provozována sdružením CESNET a Fakultou informačních technologií ČVUT. Dále se na spolupráci kolem laboratoře podílí Fakulta elektrotechnická a Institut intermédií. V roce 2014 vznikla také studentská skupina GRAFIT, která se v laboratoři i mimo ni věnuje grafické a multimediální tvorbě a popularizaci práce v laboratoři.

Zaměření a cíle laboratoře lze rozdělit do dvou skupin. První z nich - “podpora výuky” - zahrnuje popularizační přednášky z různých oborů vědy

a techniky pořádané skupinou GRAFIT a přímé zapojení laboratoře do výuky předmětů na FITu, především předmětů zaměřených na grafiku, síťové technologie a výpočty vyžadující dostatek výkonových kapacit. Z konkrétních příkladů lze uvést hodnocení a prezentace prací studentů předmětu MGA, kteří mohli využít SAGElab ke kontrole svých modelů a texturování nebo k renderování těchto modelů ve vysokém rozlišení. Do druhé skupiny - "využití laboratoře" - se řadí výzkum a vývoj technologií přímo souvisejících s laboratoří a její využití pro telekonference. Na rozvoji laboratoře se podílejí studenti v rámci svých bakalářských a diplomových prací, zaměstnanci CESNETu a také studenti v týmových projektech. Často se pak jedná o vývoj aplikací pro konkrétní události či experimenty. Více bude uvedeno v kapitole "Případy užití".

Technické vybavení

Laboratoř je vybavena telestěnou s 10K rozlišením. Konkrétně se jedná o 20 FullHD monitorů NEC instalovaných do mřížky 4x5. Stěna sestává z pěti sloupců, z nichž každý má rozlišení počtem pixelů ekvivalentní 4K rozlišení. V tomto případě je však rozlišení jedné části stěny 1980 x 4320 px (obrázek č. 0.1).

Obrázek 0.1: Telestěna

Aby bylo možné dodávat monitorům dostatečně rychle obraz v patřičném rozlišení, je každý z těchto dílů napojen na vlastní server. Vzhledem k využití laboratoře i pro renderování a jiné náročnější výpočty jsou všechny servery vybaveny 24jádrovými procesory. Propojení serverů obstarávají 10Gb optické linky. Celý cluster je pak napojen vlastní 10Gb linkou do sítě CESNETu, a umožňuje tak rychlé spojení s jinými výzkumnými institucemi po celém světě. Serverový cluster běží na Gentoo linuxu a o správu obrazovek a obsahu na nich

zobrazovaného se stará software SAGE1[3] nebo SAGE2 (viz dále). Vzhledem k probíhajícímu vývoji SAGE2 na univerzitě na Havaji a zastavenému vývoji SAGE1 není ani jedna z možností ve stavu, aby byla vyhovující pro všechny aplikace.

Kontext a koncepce projektu SAGElab

Při laboratoři SAGElab vznikla na FITu studentská skupina GRAFIT pořádající různé akce pro veřejnost i ostatní studenty. V průběhu těchto akcí začalo být jasné, že stávající ovládání telestěny, v té době ještě SAGE1, není vhodné pro použití "laickými" uživateli, a ani pro uživatele seznámeného s touto technologií není pohodlné telestěnu při přednášce takto ovládat. Následnou diskuzí členů laboratoře i GRAFITu bylo rozhodnuto, že se formou bakalářských, diplomových a semestrálních prací započne vývoj přívětivějšího GUI pro telestěnu.

Plánovaní a koncepční návrh probíhal stále ještě na systému SAGE1, který neměl v podstatě žádné rozumné GUI. V rámci předmětu BI-TUR[4] byly studenty vytvořeny grafické návrhy GUI pro mobilní aplikaci platformy Android a pro webové rozhraní (viz obrázek 0.2). Oba návrhy byly konzultovány a testovány s uživateli mimo obor IT, čímž byla získána potřebná zpětná vazba pro to, jak by mělo ovládání vypadat. Cílem bylo zpřístupnit uživateli co nejširší možnosti práce s telestěnou, ale zároveň udržet jednoduchost ovládání a v ideálním případě i jazykovou nezávislost s ohledem na pořádání mezinárodních konferencí a prezentací.

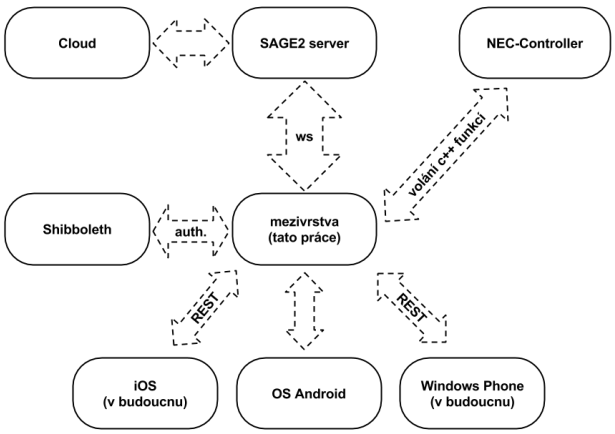
Obrázek 0.2: Návrh ovládací aplikace pro Android

[3]SAGE1 pro přehlednost a rozlišení od laboratoře a SAGE2 je takto v práci označována původní nečíslovaná verze sw SAGE (viz. seznam zkratek)

[4]bakalářský předmět Tvorba uživatelského rozhraní

Na základě těchto prací a dalších zkušeností z průběhu přednášek a používání telestěny byla vytvořena sada cílů směřujících k požadované ovladatelnosti zařízení. Z těchto cílů pak vznikla zadání bakalářských a diplomových prací. Tyto práce se zabývají vývojem datového uložiště, webového rozhraní, rozhraní pro mobilní zařízení, ovládání pomocí zařízení Kinect či vývojem specializovaných aplikací. V průběhu přípravné fáze však nastala podstatná změna, a to uvedení nové verze softwaru řídícího telestěnu. Tento nový software SAGE2 přinesl do celého projektu mnoho změn, a to jak vizuálních, tak technologických. O rozdílu technologií více v kapitole "Současný stav softwarového vybavení". SAGE2 přinesl již hotové webové rozhraní, a tak jeho tvorba, respektive přepracování, ztratilo na významu. S ohledem na technologii a vlastnosti tohoto systému se však objevila potřeba vytvoření mezivrstvy mezi systémem SAGE2 a různými ovládacími rozhraními. Struktura propojení navrhovaných aplikací je znázorněna na následujícím obrázku.

Obrázek 0.3: Koncepce projektu SAGElab



Společně by všechny součásti vyvíjené několika studenty měly vytvořit jednotné uživatelské rozhraní, které nebude plné tajemných, nic neříkajících ikon či menu. Uživatel by díky této sadě nástrojů měl být schopen snadno přecházet mezi formami ovládání. Tato práce je tedy součástí celého souboru dalších nástrojů, jejichž vývoj probíhá paralelně.

Kapitola 1

Analýza a návrh

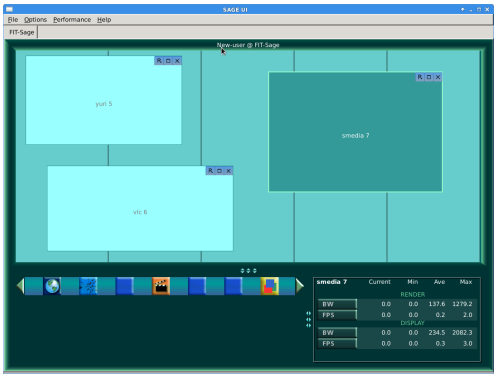
1.1 Současný stav softwarového vybavení (z pohledu uživatele)

Pokud by měl být shrnut současný stav softwaru odpovídajícího cílům této práce, pak by bylo shrnutí velmi krátké. Žádná taková mezivrstva totiž zatím neexistuje. Tato kapitola je zaměřena na celkový stav aplikací a prostředků dostupných pro využití v SAGElabu a navazuje tak na kapitolu SAGElab s tím rozdílem, že se více zaměřuje na konkrétní potřeby uživatele.

Základem prostředí, ve kterém se uživatel při práci s telestěnou pohybuje, jsou aplikace SAGE1 a SAGE2. S ohledem na zaměření práce bude kladen důraz především na druhý zmíněný.

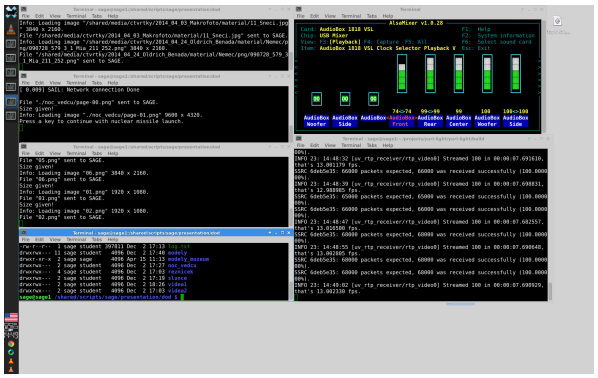
1.1.1 SAGE1

Z pohledu uživatele se jedná o jednu aplikaci, která znázorňuje rozložení oken různých aplikací na telestěně a umožňuje s nimi pohybovat a měnit jejich rozměry.



Obrázek 1.1: Screen UI SAGE1

Co se vlastních aplikací týče, je nutné být vzdáleně připojen k serveru, na kterém běží SAGE1 a ke kterému je připojena telestěna. Toto připojení probíhá přes SSH z unixové příkazové řádky. Už fakt, že se celá záležitost ovládá z příkazové řádky, je sám o sobě pro mnoho uživatelů nepřekonatelným problémem. K tomu je nutné si uvědomit, že každá aplikace běžící takto pod systémem SAGE1 se spouští přes vlastní terminál připojený přes SSH a přes něj se také ovládá. Pro uživatele tak vzniká velmi nepřehledná situace (viz. screenshot 1.2). Asi největší překážkou v použitelnosti je skutečnost, že uživatel musí věnovat nemalou pozornost právě ovládané aplikaci, místo aby se mohl plně věnovat hlavní činnosti, kvůli které přišel (např. přednášení).



Obrázek 1.2: Terminálová okna sloužící k ovládání SAGE1

Na druhou stranu nutno podotknout, že existují aplikace, které poskytují možnost ovládání přes své tradiční GUI, na jaké je jejich uživatel zvyklý z běžného desktopového použití. Jako příklad můžeme uvést VLC media player. V takovém případě pak aplikace spustí své přehrávací okno na telestěně, ale ovládací prvky přesměruje zpět přes SSH připojení, a vytvoří tak uživateli další okno na jeho počítači, z kterého ovládá aplikaci na telestěně. Toto okno pak má vzhled stejný jako tradiční verze aplikace s tím rozdílem, že například v případě VLC playeru nenívideo přehráváno v tomto okně, ale na telestěně.

1.1.2 SAGE2

Určitého pokroku v přívětivosti došlo s přechodem na SAGE2. Zásadním rozdílem je absence příkazové řádky a nutnosti přihlášení přes SSH na server. V současné době má uživatel jedinou možnost ovládání telestěny “poháněné” systémem SAGE2, a to z webového prohlížeče. Uživatel si pouze zadá adresu serveru poskytujícího aplikaci SAGE2, a tím se dostane na webovou stránku, která je rozhraním pro ovládání tohoto serveru (viz obrázek 1.3).

Jak je vidět na ilustračním screenshotu 1.3, ovládání je o poznání jednodušší a přehlednější, než bylo v případě původního SAGE1. Vše je zde na jednom místě a uživatel tak nemusí zápolit s příkazovou řádkou a hledáním

Obrázek 1.3: Webové UI SAGE2

terminálu patřícího k různým aplikacím. Interakci se systémem SAGE2 umožňuje několik ovládacích prvků využívaných v různých případech. Podívejme se tedy na jednotlivé možnosti ovládání samostatně.

- Náhledová plocha telestěny

  Usnadňuje uživateli orientaci na telestěně především v příadě, že se nenachází v její blízkosti a chce si něco připravit nebo potřebuje pohybovat s okny, aniž by rozptyloval diváky kurzorem.

- Menu SAGE2 UI

  Je realizováno formou řady ikon nacházejících se pod náhledem telestěny (Obrázek č. 1.4). Uživateli dává přístup k několika následujícím funkcím.

Obrázek 1.4: menu Webového UI SAGE2

  - Sdílení pracovní plochy, které je dostupné pouze s doplňkem do Chromu
  - Aktivaci sdílení kurzoru
  - Seznam aplikací skrytý pod ikonou rakety
  - Seznam souborů
  - Možnost spravovat uložená rozložení oken na telestěně
  - Nastavení sdíleného kurzoru pod ozubeným kolečkem, kde by se dalo spíše čekat nastavení telestěny či celého SAGE2 serveru

  - Informace skrývající základní nápovědu a přehled odkazů kde se dají nalézt výstupy SAGE2 serveru pro zobrazení živého náhledu mimo telestěnu

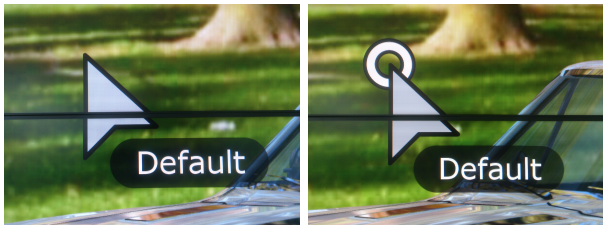
- Virtuální kurzor

  Takto bychom nazvali ovládání aplikací přímo z webového UI. Uživatel má možnost náhledová okna aplikací přemisťovat po telestěně a měnit jejich velikost. Zároveň má možnost aplikace ukončit. Celé toto ovládání probíhá uvnitř tak, že se z webového UI posílá na SAGE2 server pozice kurzoru a případná akce kurozru jako klik nebo scroll. Z toho důvodu "Virtuální kurzor", protože na telestěně, kde se aplikace pohybují, není žádný kurzor zobrazen.

  Technicky probíhá vyhodnocení pozice a akce kurzoru analýzou seznamu všech tlačítek a aktivních prvků na telestěně. Jakmile server SAGE2 dostane zprávu o akci kurzoru projde seznam všech klikatelných prvků na telestěně (aplikace, tlačítka aplikací, radial menu) a pro každé porovnává pozici s obdrženou pozicí kurzoru s ohledem na tvar a velikost konkrétního aktivního prvku. Pokud najde poziční shodu provede adekvátní akci odpovídající významu tlačítka pro které shoda nastala.

- Kurzor sdílený na telestěnu (Obrázek č. 1.5)

  Jak jsme si ukázali v popisu menu, má uživatel možnost přesměrovat si svůj kurzor na telestěnu. V tom případě se na telestěně zobrazí viditelná šipka nastavitelné barvy. Základní ovládání aplikací, jako je přesun a změna velikosti okna, jsou stejné jako v případě virtuálního kurzoru. Na rozdíl od virtuálního kurzoru má uživatel v tomto případě více možností ovládat aplikace přímo. Tuto možnost mu dává tzv. deep mode kurzoru, do kterého se dostane kliknutím pravým tlačítkem. Ve chvíli, kdy je aktivní deep mode kurozru, je interakce od uživatele přenášena přímo konkrétní aplikaci. Nejlépe je tento rozdíl vidět na aplikaci mapy nebo prohlížeč gigapanoramat. Pokud je uživatel v normálním modu kurzoru, má možnost přesouvat a zvětšovat okno aplikace. Pokud se ale přepne do deep modu, ovládá přímo aplikaci a v ukázkovém případě přibližuje mapu nebo obrázek. Ne všechny aplikace si vystačí s takto jednoduchým



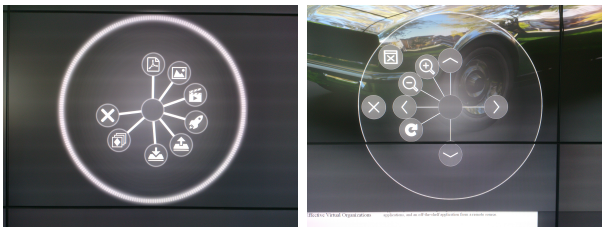
Obrázek 1.5: Kurzor v klasickém a deep modu

ovládáním jako mapy nebo obrázky.

Pro ostatní případy je zde další část ovládání zvaná Radial menu. Je to jedna z možností, jak vyřešit ovládání apliakce na telestěně. Druhou zde nevyužitou možností by bylo nechat ovládací prvky aplikací stále viditelné, tak jak jsme na to zvyklí z běžného desktopového prostředí. Pro využití na telestěně je to ale docela zbytečné plýtvání zobrazovací plochou. Autoři SAGE2 proto zvolili první variantu a tou je tzv. Radial menu na které se podíváme blíže.

- Radial menu

  Ovládací prvek zpřístupňující speciální funkce aplikací na telestěně a v případě, že není ovládána konkrétní aplikace, umožňuje i přístup k menu SAGE2. Existují tedy dvě verze (viz. obrázek č 1.6). První je jen jiným zobrazením menu SAGE2 a není nutné ji blíže popisovat. Zajímavější je druhá varianta zpřístupňující ovládání apliakcí. Na obrázku je vidět příklad, jak může pro konkrétní aplikaci vypadat. Obvykle umožňuje posun na další soubory v případě prohlížečů či možnost otevřít v dané aplikaci nějaký soubor. U videopřehrávače pak obsahuje klasické ovládání, které by uživatel čekal.



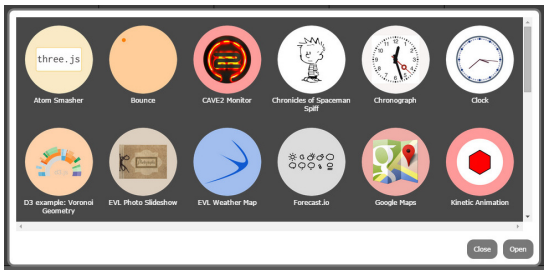
Obrázek 1.6: Aplikační a obecná verze Radial Menu

  Radial menu je přístupné pod pravým tlačítkem, což do jisté míry minimalizuje nepraktičnost způsobenou skrytím ovládání před zraky uživatele tím, že uživatelé jsou na kontextová menu pod pravým tlačítkem zvyklí.

  Nevýhodou Radial menu je častá chybovost a nepřesnost kurzoru. Dost často se stává, že pokud si uživatel spustí radial menu, začne mu před kurzorem utíkat a nelze se ho zbavit. Nepřesnost kurzoru pak způsobuje problémy s ohledem na malou velikost ikon Radial menu.

Kromě zmíněných ovládacích prvků SW SAGE2 nabízí celou řadu aplikací upravených pro použití na telestěně. Rozbor jadnotlivých aplikací není předmětem této práce, proto jen stučné shrnutí. Jsou zde použitelné aplikace jako mapy, prohlížeč webGL modelů nebo prohlížeč obrázků a pdf. Dále je zde zajímavá apliakce na gigapanoramata, která ovšem umí otevřít jediný soubor,

který je napevno určený v jejím zdrojovém kodu [3]. Další aplikace buď nefungují vůbec nebo jsou díky svému vývojovému stadiu v praxi nepoužitelné. Přehled aplikací je k vidění na obrázku č 1.7.



Obrázek 1.7: Přehled aplikací dostupných v menu SAGE2

Ve výsledku se tedy u SW SAGE2 jedná o pokrok a pro uživatele je celé použití snazší. Na druhou stranu je nutné si uvědomit, že ani tak není u složitějších operací plně intuitivní a bez zaškolení se uživatel neobejde. Pro prezentace je pak hlavní nevýhodou stále přetrvávající nutnost ovládání z desktopového OS kvůli nepřítomnému rozhraní pro mobilní platformy.

1.2 Případy užití

Hned od počátku je třeba rozlišovat využití telestěny a vyvíjené mezivrstvy. Přestože se práce zabývá návrhem a implementací mezivrstvy, je pro porozumění tématu nutné zmínit i různé případy využití pro telestěnu a SAGElab jako takový. Využití telestěny je následující:

- prezentace a přednášky

  V současné době nejčastější využití laboratoře. Jsou zde pořádány zájmové přednášky z cyklu Grafické čtvrtky a příležitostně se zde koná výuka.

- telekonference

  Připojení SAGElabu k internetu vysokorychlostním připojením jej přímo předurčuje k využití pro on-line konference či jiné formy spolupráce na dálku. Je také možné využít zrcadlení obsahu telestěny, což zjednodušuje spolupráci týmů z opačných konců světa.

- výzkum využití HiRes technologií a materiálů

  - zkoumání materiálů z jiných oborů v HiRes

    Hosté Grafických čtvrtků projevili zájem o přístup k telestěně za účelem zkoumání vlastních materiálů pořízených například mikroskopem či jinými vědeckými přístroji.

  - experimenty s prezentací fotografií pořízených technologií Gigapan[5]
  - využití různých interakčních technologií k prezentaci materiálů v HiRes

- výzkum a vývoj síťových technologií
  - rozvoj laboratoře a telestěny
  - využití pro demonstrační a vědecké živé přenosy (koncerty, operace,...)

- napojení na CHUL[6]

  V průběhu letního semestru 2015 by mělo dojít k propojení SAGElabu s Children usability laboratoří, která se bude zaměřovat na user interface design aplikací pro dětské uživatele. Konkrétní představa propojení je zatím ve fázi příprav a analýzy, ale je třeba s ní počítat při vývoji.

Do budoucna je nutné při vývoji nových aplikací a rozvoji těch stávajících brát ohled na tyto potřeby. Důležitá pak je především skutečnost, že výhledově by kterýkoliv ze zmíněných případů užití měl být schopen zajistit si sám uživatel po krátkém zaškolení ohledně specifik zařízení. V ideálním případě pak i bez tohoto zaškolení.

Nyní k otázce, jaké případy využití z toho vyplývají pro navrhovanou mezivrstvu. Jedná se v podstatě o uživatelské "klienty", sloužící k snadnému a rychlému ovládání telestěny a jejích přidružených součástí. Klienty alias uživatelskými aplikacemi jsou následující UI.

- mobilní aplikace (android, iOS, Windows Phone)

  Ovládání pomocí aplikace z tabletu nebo chytrého telefonu je pro většinu uživatelů běžnou záležitostí, proto dává smysl přiblížení telestěny uživateli tímto způsobem. Konkrétní představa ovládání je popsána v samostatných pracích.

- další formy ovládání, jejichž vývoj zatím nebyl zahájen

  Možnost ovládání telestěny nově vznikajícími zařízeními, jako jsou Ring controler[7], Leap motion controler[8] nebo již dobře známý Kinect. Jedná se o výhled do budoucna především z toho důvodu, že uvedené technologie jsou často v raných vývojových stádiích a jejich spolehlivost je nejistá.

[5]Gigapan: technologie pro vytváření a prezentaci fotografií v rozlišení řádově stovek MPix a více viz: http://gigapan.com

[6]Children usability lab: laboratoř pro testování uživatelské použitelnosti se zaměřením na děti

[7]Ring Controller: prsten sloužící k bezdrátovému ovládání mobilních zařízení viz: www.logbar.jp/ring/en

[8]Leap Motion Controller: zažízení podobné Kinectu viz: www.leapmotion.com

Možných způsobů využití je tedy mnoho a jak je vidět, jsou i dost různorodé. Již dříve však bylo zmíněno, že cílem je zpřístupnění telestěny pro každodenní použití, a tomu se podřizuje i zapracování různých využití do vyvíjených aplikací. Pro začátek se například provádí vývoj mobilní aplikace pouze na platformě Android a ovládání pomocí zmíněných gadgetů bude pravděpodobně umožněno později.

1.3 Proč mobilní aplikace a mezivrstva

Kapitola zabývající se využitím laboratoře ukazuje, že uživateli zařízení by měli být, kromě programátorů a vývojářů, i lidé z jiných oborů než je IT. Právě pro takové uživatele je přirozenější ovládat telestěnu pomocí webového prohlížeče nebo tabletu. Jak už bylo zmíněno v kapitole SAGE2, jistá verze webového rozhraní je již součástí tohoto systému. Jak se ukázalo při zkoumání a testování systému SAGE2, má toto rozhraní dva základní nedostatky; prvním jsou jeho velmi omezené možnosti a někdy jejich dost neintuitivní chování či nelogické umístění v položkách menu. Druhou, a pro účel této práce důležitější nevýhodou je pak nepoužitelnost na mobilních zařízeních, např. typu Android. V průběhu testování bylo zjištěno, že i když je ovládací web responzivní co se týče vzhledu, jeho funkčnost již tak dobrá není.

Konkrétně se jedná například o nefunkčnost ovládání kurzoru, který v tomto webovém rozhraní nereaguje na klikání pomocí dotykové obrazovky. S ohledem na rozdílné chování aplikací v mobilním prohlížeči a těch, které jsou napsány jako nativní aplikace, bylo nakonec rozhodnuto zvolit možnost nativní aplikace. Více o této aplikaci v samostatné bakalářské práci (BP). Mezivrstva, jejíž tvorbou se zabývá tato práce, jistě se můžeme ptát, zdali je vůbec k něčemu užitečná. Tato otázka je jistě na místě, ale hned si ukážeme, že i vývoj této mezivrstvy je smysluplný.

Systém SAGE2 je stále ve vývoji a probíhají v něm změny. Z toho důvodu je pro účely vývoje připojených aplikací v SAGElabu výhodnější mít jedno přípojné místo a nutné úpravy v kódu tak v budoucnu dělat pouze na této mezivrstvě, a ne na všech aplikacích pro různé platformy. Mezivrstva dále umožní vytvoření funkčního rozhraní pro aplikace systému SAGE2, které ještě nejsou implementovány. Další neméně důležitou výhodou oproti přímému napojení ovládací aplikace na SAGE2 je to, že je nutné napojení i na další dílčí aplikace související s použitím telestěny. Jedná se konkrétně o software managera obrazovek, který, jak je z požadavků na mezivrstvu patrné, bude ovládat fyzické monitory tak, aby bylo možné je na dálku zapnout a upravovat jejich vlastnosti např. jas. V průběhu diskuze nad celkovým konceptem ovládání pak vyplynula ještě nutnost napojení datového uložiště do této mezivrstvy, aby měl uživatel ovládacích aplikací možnost přistupovat ke svým souborům.

Mezivrstva zároveň vytváří možnost ověřovat identitu uživatele na jednom místě pomocí autorizačního tokenu od frontend aplikací.

Na základě uvedených výhod bylo rozhodnuto o vytvoření aplikačního rozhraní ve formě této mezivrstvy tak, aby další práce byly co nejvíce odstíněny od vývoje systému SAGE2, který není pod kontrolou laboratoře. Ačkoliv mezivrstva přináší další kód do projektu, v němž může nastat problém, jsou výhody fixního rozhraní pro stěnu jednoznačné.

V následující kapitole shrneme, jaké konkrétní požadavky jsou na připravovanou mezivrstvu kladeny ze strany mobilních aplikací, potažmo uživatelů, a ze strany nutného napojení na laboratoř a práce ostatních členů vývojového týmu.

1.4 Funkční požadavky (FP)

1. rozhraní poskytuje následující funkce pro ovládání HW telestěny v SAGE-labu
   - 1.1. zapnutí a vypnutí každého monitoru zvlášť i všech najednou
   - 1.2. podpora skupin monitorů
   - 1.3. nastavení jasu pro každý monitor či pro všechny najednou
2. rozhraní poskytuje možnosti práce s okny a aplikacemi spuštěnými na telestěně
   - 2.1. poskytuje seznam běžících aplikací s jejich umístěním na telestěně
   - 2.2. umožňuje spustit požadovanou aplikaci a otevře její okno
   - 2.3. poskytuje funkce pro přemístění oken na telestěně
   - 2.4. poskytuje funkce pro změnu velikosti oken na telestěně
   - 2.5. poskytuje možnost zavřít konkrétní okno aplikace či všechna běžící okna
3. rozhraní poskytuje možnost přímého ovládání kurzoru na telestěně na vyžádání
   - 3.1. umožňuje vytvořit (připojit) kurzor
     Bude existovat funkce, která spustí zobrazování kurzoru na SAGE telestěně.
   - 3.2. umožňuje nastavit barvu a pojmenování kurzoru
   - 3.3. umožňuje pohybovat kurzorem podle vstupu udávajícího posunutí
   - 3.4. poskytuje informaci o aktuální poloze kurzoru
4. rozhraní poskytuje možnost práce se soubory
   - 4.1. na základě napojení na rozhraní file managera poskytuje seznam souborů odpovídajícím zadaným parametrům (vlastník, typ souboru)
   - 4.2. poskytuje možnost volání funkce otevři_soubor(id)
5. podpora autentizace přes shibboleth
   - 5.1. ověřování pravosti obdrženého tokenu uživatele vůči poskytnuté autoritě

1.5 Nefunkční požadavky (NP)

1. aplikace poskytuje REST API pro mobilní aplikace
   1.1. propojení s klientem pro platformu Android vyvíjené v rámci BP Datem Pham Tatem[1]
2. nasazení v SAGElabu
   2.1. aplikace běží na linuxových serverech v laboratoři SAGElab
   2.2. aplikace je napojená na systém SAGE2
   realizace pomocí Web socketů
   2.3. aplikace je napojená a systém ovládající monitory telestěny
   propojení s API aplikace vyvíjené v rámci BP Patrikem Faistaverem[2]
3. podpora více připojených klientských aplikací
4. zveřejnění aplikace pod licencí CC (přesněji podle licencí sage commons)[18]
5. dokumentace bude v Angličtině
6. dodržení "štábní kultury"

1.6 Možnosti napojení na telestěnu

1.6.1 API poskytované sw SAGE2

Komunikace se softwarem SAGE2 probíhá pomocí technologie websocketů. Rozhraní poskytované napojovaným aplikacím je díky tomu založené na asynchronní komunikaci. Pro každého klienta je vytvořeno vlastní websocketové spojení, kterým pak probíhá výměna zpráv. Klientem je v tomto případě myšlena každá aplikace, která nějakým způsobem má ovládat, co se děje na telestěně řízené SW SAGE2. Formát Websocketových (ws) zpráv je obecně následující (ukázka 1.1). Posílají se data ve formátu JSON, která obsahují nějaký název požadavku a dále pole dat nutných k vykonání požadavku. Seznam všech relevantních zpráv a jejich popis je uveden v digitální příloze *"WebSocketové zprávy systému SAGE2"*.

Listing 1.1: Základní formát WS zprávy

```
{
  "f": "messageName",
  "d": {
    "messageData"
  }
}
```

V průběhu vytváření websocketového spojení je nutné z klientské aplikace poslat inicializační zprávu, která obsahuje parametry komunikace. V ukázce č. 1.2 je vidět formát této zprávy. SAGE2 podle tohoto nastavení posílá požadované informace.

Listing 1.2: Inicializační ws zpráva pro SAGE2

```
{
  "f": "addClient",
  "d": {
    "clientType": "sageUI",
    "sendsPointerData": true,
    "sendsMediaStreamFrames": true,
    "uploadsContent": true,
    "requestsServerFiles": true,
    "sendsWebContentToLoad": true,
    "launchesWebBrowser": true,
    "sendsVideoSynchonization": false,
    "sharesContentWithRemoteServer": false,
    "receivesDisplayConfiguration": true,
    "receivesClockTime": false,
    "requiresFullApps": false,
    "requiresAppPositionSizeTypeOnly": true,
    "receivesMediaStreamFrames": false,
    "receivesWindowModification": true,
    "receivesPointerData": false,
    "receivesInputEvents": false,
    "receivesRemoteServerInfo": false
  }
}
```

Z přiložené ukázky (1.2) jsou dobře vidět možnosti, jaké rozhraní SAGE2 přes websockety poskytuje. Nastavení funguje tak, že položky "send. . . " dávají SAGE2 na vědomí, že přihlašující se klient bude posílat tyto informace, respektive že bude žádat provedení tohoto typu operací. "Receive.." zprávy pak nastavují na SAGE2, co vše má této aplikaci posílat za informace. SAGE2 pak na základě této inicializační zprávy nastaví své rozhraní tak, aby zbytečně nezatěžovalo linku daty, která nikdo nevyužije. Jako příklad lze uvézt parametr *"uploadsContent": true*, který určitě nebude nastaven na "true", pokud půjde o ovládací aplikaci napojenou kupříkladu na Kinect, nebo jiné podobné zařízení. V opačném směru, tedy od SAGE2 ke klientské aplikaci, putují v průběhu inicializace zprávy obsahující inicializační hlavičku, konfiguraci obrazovek telestěny, informaci o verzi HW SAGE2 a případně zprávy předávající informace o aktuálně spuštěných aplikacích na SAGE2 serveru. Z ukázky č. 1.3 kodu zprávy je vidět, jaké informace o konfiguraci telestěny se klientská aplikace dozví. Jedná se především o rozlišení jednotlivých monitorů, z kterých je telestěna složená, dále její konfigurace, tedy jak jsou monitory umístěné a kolik jich celkem je. Ukázka je zkrácena o informace o verzi HW a informaci o tom, jaké nastavení barvy má pozadí telestěny, případně jaký obrázek na něm je použit.

Listing 1.3: zpráva obsahující konfiguraci SAGE2

```
{
  "f": "setupDisplayConfiguration",
  "d": {
    "name": "bp-sage",
    "host": "195.113.232.57",
    "port": 9494,
    "index_port": 8484,
   ....
    "resolution": {
      "width": 1920,
      "height": 1080
    },
    "layout": {
      "rows": 2,
```

```
      "columns": 2
    },
    "displays": [
      {
        "row": 0,
        "column": 0
      },
  ....
    ],
    "totalWidth": 3840,
    "totalHeight": 2160,
  ....
  }
}
```

Po navázání spojení a předání počátečních dat nutných k nakonfigurování klienta již následuje běžná výměna zpráv. Klient posílá dva druhy zpráv. První druh se týká konkrétních požadavků na nějakou akci s oknem či aplikací. Například se jedná o požadavky na poskytnutí seznamu aplikací či dostupných souborů. Do této kategorie zpráv patří také požadavky na spuštění položek z těchto seznamů. Viz ukázka (č. 1.4) zprávy posílané při požadavku na spuštění aplikace *"clock"*. Na zprávě je dobře vidět také dříve zmíněný formát obsahující název požadavku následovaný daty nutnými k jeho vykonání.

Listing 1.4: Spusť aplikaci

```
{"f":"loadApplication",
        "d":{"application":"clock","user":"195.113.232.34:54267"}}
```

Druhá skupina zpráv se týká kurzoru. Pro mnoho akcí SAGE2 využívá informaci o pozici kurzoru a informaci o tom, kolikrát bylo kliknuto, a teprve na své straně analyzuje, na co uživatel kliknul a jakou má provést akci. Jak je vidět na procesu, jakým SAGE2 zpracovává obyčejný dvojklik, je tento přístup často nepraktický. Je sice pravda, že UI aplikace se tak nemusí starat o zjišťování, na co uživatel klikl, na druhou stranu, konkrétně v případě mobilních aplikací, tento přístup nevyhovuje. Jak vyplývá z analýzy požadavků na uživatelské rozhraní mobilní aplikace pro platformu Android, je lepší mít ovládací prvky přímo na tabletu, přizpůsobené specifikům dotykového ovládání a malé obrazovky. S ohledem na fakt, že tato mezivrstva má sloužit primárně pro mobilní platformy, bude lepší udělat lehké úpravy v rozhraní SAGE2 tak, aby podporoval ovládání aplikací pomocí přímého volání funkcí. To znamená, že například v současné době maximalizace okna realizovaná na základě vyhodnocení dvojkliku (viz ukázka č. 1.5), bude nahrazena websocketovou zprávou *"makeWindowFullScreen"*. Díky těmto úpravám, jejichž konkrétní podoba je popsána v kapitole o realizaci, bude možné si v mobilní platformě, využívající tuto mezivrstvu, vytvořit UI podle vlastních potřeb, a to pak napojit na ovládání aplikací.

Listing 1.5: Vyhodnocení dvojkliku

```
{"f":"pointerPosition","d":{"pointerX":369.71..,"pointerY":126.039..}}
{"f":"pointerPress","d":{"button":"left"}}
{"f":"pointerPosition","d":{"pointerX":369.71..,"pointerY":126.039..}}
{"f":"pointerRelease","d":{"button":"left"}}
```

```
{"f":"pointerPosition","d":{"pointerX":369.71..,"pointerY":126.039..}}
{"f":"pointerPress","d":{"button":"left"}}
{"f":"pointerPosition","d":{"pointerX":369.71..,"pointerY":126.039..}}
{"f":"pointerRelease","d":{"button":"left"}}
{"f":"pointerDblClick"}
```

Na základě obdržených zpráv SAGE2 provede požadované změny na telestěně a podle toho, co si připojení klienti nastavili v konfiguraci svého spojení, je všem klientům rozeslána aktualizace stavu zobrazení oken na telestěně a stavu SAGE2 serveru. Na základě těchto zpráv si každá klientská aplikace musí aktualizovat svojí vlastní reprezentaci telestěny. Připojení více klientských aplikací je řešeno na úrovni websocketů, kdy SAGE2 server udržuje spojení s každým klientem, identifikované jeho adresou a portem. Pro rozlišení více klientů připojených přes mezivrstvu, jež je předmětem této práce, pak stačí udržovat seznam ws spojení a na jeho základě obstarávat komunikaci.

1.6.2 Knihovna NEC-Controller

S ohledem na FP1.x je nutné zapracovat do ovládání také rozhraní pro řízení monitorů. Vlastní kontrolu nad těmito monitory má aplikace vyvíjená Patrikem Faistaverem v rámci jeho Bakalářské Práce [2].

Na základě požadavků z analýzy BP *"Android manager zařízení SAGE"* a možností nabízených zmíněnou knihovnou, bude základní ovládání zahrnuto do jednotného REST API této mezivrstvy. Využití C++ knihovny umožní knihovna node-ffi [**?**]. Použití znázorňuje ukázka č. 1.6

Listing 1.6: Import knihovny ffi

```
var faistapat = ffi.Library('./libs/controller.so', {
  'initController': [ "void" , [] ],
  'addMonitor': [ "void", [ "string", "int", "int" ] ],
  'connectAll': [ "bool", [] ],
  ...
  'destroyController': [ "void", [ "void" ] ],
  'powerStatusRead': [ "int", [ "int" ] ]
});
```

Knihovna ovládající monitory poskytuje funkce pro pokrytí většiny požadavků FP1.x. Lze tedy ovládat každý monitor zvlášť, a to co se týče zapínání i nastavení jeho jasu a dalších vlastností. Pro ovládání přes tuto mezivrstvu sloužící pro mobilní aplikace bude ovšem z nastavení zpřístupněno jen ovládání jasu. Systém je tak ochráněn před náhodným zásahem uživatele do kalibrace barev monitorů. Z požadavků FP1.1 až FP1.3 ovšem vyplývá, že by měl být podporován přístup i ke všem monitorům najednou nebo k jejich skupinám. Tato funkčnost není knihovnou podporována, a proto bude implementována v rámci mezivrstvy pomocí opakovaného volání funkcí na ovládání jednotlivých monitorů. Posledním požadavkem z první skupiny je ovládání audiovýstupů telestěny. Knihovna k monitorům sice umožňuje ovládat jejich hlasitost, audio výstupy monitorů ovšem nejsou připojené k žádnému zdroji audio signálu a jsou tak nevyužitelné. Realizace tohoto požadavku tedy není v současné době

možná a vyžadovala by zásah do dalších aplikací souvisejících s telestěnou tak, aby bylo přístupné ovládání hlasitosti výstupů ze zvukové karty.

Práce s knihovnou pro ovládání monitorů pak probíhá následujícím způsobem. (všechny následující ukázky v podkapitole jsou v C++ tedy bez includu do javascriptu)

```
void initController();
void addMonitor(const char * Addr, int port, int ID);
bool connectAll();
```

Jak je vidět, knihovna vyžaduje prvotní inicializaci a následně přidání monitorů včetně jejich IP adres. Následně je možné připojit všechny monitory najednou. Pokud se připojení povede, jsou už monitory připravené k ovládání. Identifikace monitorů pak probíhá pomocí ID, které bylo při jejich přidávání zvoleno.

```
int  getBacklight       (int ID);
void setBacklight       (int ID, int val);
```

Pro nastavení podsvícení je k dispozici funkce sdělující aktuální stav a pak nastavovací funkce, jejímž parametrem je kromě ID také hodnota, na jakou chceme podsvícení nastavit, a to v rozsahu 0 - 100 %.

```
int powerStatusRead(int monitorID);
void powerControl(int monitorID, int powerMode);
```

Funkce ovládající zapnutí / vypnutí monitoru jsou stejně jako všechny ostatní, které nejsou využité, analogické svým chováním k již zmíněnému podsvícení. Zapínací funkce však kromě poloh zapnuto a vypnuto disponuje ještě polohami Stand-by a Suspend. Celkem tak poskytuje 4 stavy každého monitoru[9].

```
void disconnectAll     ();
void destroyController();
```

Na závěr je pak opět nutné monitory odpojit a odpojit i celý ovládací program. Knihovna samozřejmě poskytuje i další funkce pro nastavení ostatních parametrů. S ohledem na to, že pro ovládání z tabletu nejsou využity, nejsou zmíněny ani v této práci. V případě potřeby jsou k nastudování v BP Patrika Faistavera [2].

1.7 Struktura aplikace a její napojení na SAGE

Aplikace je vyvíjená jako mezivrstva a sjednocující rozhraní pro ovládání SAGE2 a telestěny z mobilních zařízení a jiných UI zařízení. Tato skutečnost

[9] 1: ON | 2: Stand-by (power save) | 3: Suspend (power save) | 4: OFF (same as IR power off)

je klíčová pro vlastní návrh struktury propojení. Je zde nutné propojit HW SAGE2, který se stará o zobrazování obsahu na telestěně, dále jeho správu souborů, která bude v budoucnu nahrazena vlastním cloudovým řešením. Ze strany HW je zde propojení s knihovnou NEC-Controller a směrem ke koncovým zařízením pak aplikace poskytuje REST API.

Komunikace s aplikací SAGE2 probíhá pomocí technologie ws. Bližší popis, jak komunikace vypadá, je popsán v kapitole SAGE2, přehled všech využívaných zpráv je v příloze Websocketové zprávy SAGE2. Aplikace si udržuje jedno spojení pomocí ws se SAGE2 serverem pro získávání informací ohledně stavu aplikací a oken. Kromě tohoto cenrátlního ws spojení umožňuje vytvoření spojení pro každého klienta, které je využité pro ovládání kurzoru.

Ovládání monitorů je zajištěno za pomoci knihovny NEC-Controller. Knihovna po naimportování do javascriptové aplikace poskytuje klasické volání funkcí, které poskytují přímou síťovou komunikaci s monitory.

Posledním článkem aplikace je poskytované REST rozhraní. Pomocí něj se mobilní aplikace mohou dotazovat na aktuální stav nebo požadovat jeho změnu. Podrobněji bude toto popsáno v kapitole "Návrh poskytovaného API". Na základě analýzy možností REST technologie bylo zjištěno, že pro využití v takovémto asynchroním prostředí má jisté nevýhody. Jedná se především o fakt, že nepodporuje předání informací bez dotazu. S ohledem na tuto skutečnost bude do aplikace přidána ještě ws část API, která bude v případě kompatibilního klienta poskytovat změny na SAGE2 pomocí ws zpráv. Především se jedná o změnu velikosti a pozic oken, kterou může vyvolat i jiný klient. V případě klienta, který ws nemá implementovaný, bude nutné z jeho strany provádět opakovaně dotazy na tuto informaci přes příslušný REST request.

S ohledem na rozdílnou charakteristiku ws a REST komunikace bude většina informací ze SAGE2 ukládána v mezivrstvě a připravena na odeslání v případě dotazu ze strany REST klienta. Informace nutné k aktualizaci stavu v klientské aplikaci budou posílány ws zprávou všem připojeným klientům tak, aby se aktualizace předávaly co nejrychleji. Požadavky související s monitory jsou pak prováděné standardně ve chvíli volání.

Takováto realizace má jistě své nevýhody, především se jedná o možnost získání neaktuálních dat v určitých případech. To by se dalo řešit čekáním na aktualizaci dat v mezivrstvě v každém požadavku ze strany klienta, ovšem způsobovalo by to zpomalení a prodloužení času čekání na odpověď k požadavku. S ohledem na nízkou míru výskytu této situace bylo rozhodnuto, že není nutné se tímto aktuálně zabývat, a celé řešení tím zpomalovat. V případě, že by se při uživatelském testování zjistil vysoký výskyt chyb způsobených neaktuálními daty, je možné toto upravit.

1.8 Návrh poskytovaného API

Poskytované API bude fungovat na principu REST technologie doplněné o ws v případech, kde není využití RESTu vhodné. Konkrétní případy vyplývají z analýzy chování SAGE2 a možností technologie REST. V rámci kapitoly implementace bude upozorněno na konkrétní výskyty tohoto rozšíření.

REST technologie poskytuje možnosti pro přístup ke zdrojů serveru prostřednictvím 4 typů požadavků, a to GET, PUT, POST a DELETE. Jedná se o bezestavovou komunikaci klient-server. To znamená, že server si neudržuje žádné informace o stavu a klient tak musí dodat vše potřebné, aby mohl dostat odpověď na svůj dotaz. Tradičně se REST API využívá v aplikacích pracujících s databází, kde zpřístupňuje SQL dotazy na úpravy DB. V případě této mezivrstvy je využití poměrně odlišné, a proto i návrh může být v některých ohledech neintuitivní. Dále bude pospáno využití jednotlivých typů požadavků, možné návratové kody těchto požadavků a základní rozdělení URL podle kategorie zdroje, na který je dotaz směřován. Na závěr bude vysvětleno rozšíření o ws zprávy.

1.8.1 Metody

- GET

  Bude zpřístupňovat informace a data poskytovaná serverem SAGE2 a HW telestěny. Klientská aplikace tak bude mít přístup k informacím o konfiguraci telestěny po SW i HW stránce. Dále bude umožňovat získávání seznamů dostupných apliakcí a souborů. Zároveň bude sloužit k získávání aktuálního stavu zobrazených aplikací na telestěně. (viz podkapitola o ws rozšíření)

- POST

  Na rozdíl od běžně zažitého standardu nebude nic vytvářet. Místo toho bude umožňovat spouštět zdroje. Předně se jedná o zapínání monitorů telestěny a otevírání aplikací a souborů. Dále také bude zapínat sdílení kurzoru.

- PUT

  Bude sloužit k aktualizaci zdrojů, respektive umožní klientské aplikaci upravovat nastavení monitorů. Dále bude umožňovat ovládání kurzoru a aplikačních oken. Rozsah možných akcí vychází z FP. Kromě plně funkčního kurzoru se jedná hlavně o možnost přemisťovat okna, a také je přímo ovládat na základě ovládacích prvků příslušné aplikace.

- DELETE

Opět, na rozdíl od obvyklé DB praxe, nebude nic mazat. Bude umožňovat vypínání monitorů a zavírání běžících oken. Též bude sloužit k ukončení sdílení kurzoru.

1.8.2 Návratové kody

Seznam návratových kodů je samovysvětlující až na rozdíl mezi 400 a 501. V případě 401 se jedná o chybu, kdy klientská aplikace požadovala zdroj, který není k dispozici, protože se jedná o nesmyslný požadavek, nebo jeho realizace není možná. Naproti tomu 501 bude vracen v případě, že bude přijat relevantní požadavek. Z důvodu probíhajícího vývoje dalších součástí systému ovládání SAGE2 a samotného serveru SAGE2 není implementace hotová, ale do budoucna je plánována. Seznam všech návratových kodu je následující:

- 200 OK
- 400 Unsupported request
- 401 Unauthorized
- 404 Not found
- 501 Not implemented
- 500 Internal server error

1.8.3 Routování

API bude na základě zjištěných FP poskytováno v několika základních kategoriích URL. Každá kategorie představuje určitou logickou oblast ovládání. URL samotné jsou svým názvem samovysvětlující. Dále je uveden jejich seznam s krátkým popisem.

- /wall - ovládání telestěny
- /monitors - ovládání monitorů
- /apps - přístup k aplikacím instalovaným na SAGE2 serveru
- /files - přístup k souborům dostupným na SAGE2 serveru nebo na připojeném uložišti
- /appWindows - ovládání běžících oken aplikací na telestěně
- /pointer - ovládání vzdáleného kurzoru myši

1.8.4 Rozšíření o ws message

S ohledem na charakter technologie REST, která neumožňuje asynchronní předání aktualizace od serveru směrem ke klientovi, bude implementovaná podpora websocketů. Tato websocketová komunikace nebude poskytovat rozhraní v plném rozsahu, ale pouze v kritické části aplikace, kde je tento typ komunikace vyžadován. Implementace tedy bude zahrnovat ws rozesílání aktuálního stavu zobrazených aplikací na SAGE2. V případě chybějící podpory pro ws ze strany klienta bude stejný formát aktualizace poskytovatelný i přes REST API. V tomto případě se pak počítá s periodickými dotazy na aktualizaci ze strany klientské aplikace.

Kapitola 2

Realizace

2.1 Poskytované REST API

Podle všeobecného návrhu komunikačního rozhraní mezivrstvy byla realizována sada REST API funkcí pokrývajících funkční požadavky vzešlé z analýzy potřeb uživatelů. Popis rozhraní je členěn podle typu REST požadavku a dále pak podle skupiny FP, kterou realizuje. V příloze na CD je vygenerovaná kompletní dokumentace API[10] pro lepší orientaci při jeho dalším využití. REST API bylo realizováno pomocí node.js knihovny express [4]. Po zvážení výsledků analýzy bylo rozhodnuto navržené REST API doplnit o zmíněné ws zprávy. Popis jejich implementace je zařazen na vhodná místa této kapitoly, přestože se nejedná o součást REST technologie, ale přímo souvisí s REST požadavky, ke kterým byla přičleněna.

2.1.1 GET

Jak vyplývá z návrh a zvyklostí používání, jsou GET requesty využity pro získání informací ze SAGE2 serveru. Konkrétně se pak jedná o informace o telestěně jako takové, tedy jejím HW, běžících aplikacích a dostupných zdrojích.

FP1 První skupina požadavků se týkala práce s telestěnou a jejím HW. Pomocí GET requestů jsou zpřístupněny informace o monitorech telestěny a konfiguraci sw SAGE2 tak, aby bylo možné nastavit UI na mobilní aplikaci.

URL '/wall' (ukázka 2.1) zpřístupňuje klientské aplikaci konfiguraci SAGE2. Uživatel je na základě poskytnutého tokenu ověřen a v případě, že má oprávnění, je mu poskytnuta odpověď obsahující informace ve formátu JSON. (viz ukázka 2.2) Data jsou do proměnné SAGEConfiguration uložena

[10] dostupné z: http://webdev.fit.cvut.cz/ bremondr/BP-doku/ApiDoc/

na základě zpráv příjmaných přes ws ze SAGE2 serveru v průběhu inicializace ws spojení.

Listing 2.1: GET(wall)

```
app.get('/wall', function(req, res){
  if(isTokenValid(req.body.token)){
     res.type('json');
     res.send(SAGEConfiguration);
  } else {
     res.type('text/plain');
     res.statusCode = 401;
     res.send("Error 401: You are not authorized...");
  }
});
```

Listing 2.2: JSON response for GET(wall)

```
{
  "name": "bp-sage",
  "host": "195.113.232.57",
  "port": 9494,
  "index_port": 8484,
  "display_resolution": {
    "width": 1920,
    "height": 1080
  },
  "layout": {
    "rows": 2,
    "columns": 2
  },
  "totalWidth": 3840,
  "totalHeight": 2160
}
```

URL '/monitors' ('/monitors/:id') umožňuje získávat informace o stavu monitorů. Autoizovanému uživateli sestaví opakovaným voláním knihovny NEC-Controller JSON odpověď obsahující informace o všech monitorech (o vybraném monitoru). Informace o všech monitorech jsou předány polem obsahujícím jednotlivé monitory (ukázka 2.4). Ke každému monitoru je poskytnuta informace, jestli je zaplý, a v případě, že ano, tak i hodnota podsvícení.

Listing 2.3: GET(monitors)

```
  for( var i = 0; i < 20; i++){
      status    = faistapat.powerStatusRead(i+1);
      if(status == 1)
         backlight = faistapat.getBacklight(i+1);
       else
          backlight = 0;
       re.push({"id": i+1, "status": status, "backlight": backlight});
  }
```

Listing 2.4: JSON response for GET(monitor)

```
{
  "id": "1",
  "status": 1,
  "backlight": 50
}
```

FP2 Druhá skupina požadavků se týká práce s okny a aplikacemi na SAGE2. GET requesty v tomto případě realizují získávání informací o dostupných aplikacích a o aplikacích aktuálně běžících.

URL '/apps' poskytuje seznam dostupných aplikací. Ten je v průběhu inicializace zjednodušen jen na nutné informace a uložen do proměnné na straně mezivrstvy, na GET request je pak vydán jako JSON seznam názvů aplikací.

URL '/appWindows' umožňuje přístup k seznamu aplikací běžících na SAGE2 serveru. Tento seznam je průběžně aktualizován na základě zpráv posílaných přes ws ze serveru. Realizace je popsána v následující podkapitole "Webscoket". Tento GET request je také místem, kde se projevuje nevhodnost využití REST technologie pro některé části tohoto řešení. Klientská aplikace má dvě možnosti, jak se s tímto vypořádat. První z nich je volání tohoto GET requestu opakovaně v nějaké poměrně vysoké frekvenci, což zbytečně zatěžuje síť, a především v případě mobilních zařízení, u nichž se předpokládá bezdrátová komunikace, není toto zatížení vhodné. Proto byla do REST rozhraní přidána webscoketová část, která při každé aktualizaci seznamu běžích aplikací, ať už jejich přesunu či zvětšení okna nebo změny seznamu běžících aplikací, pošle ws zprávu s aktualizovaným seznamem všem klientům připojeným přes ws. Formát odpovědi přes REST i zprávy z ws je pak stejný. (viz. ukázka 2.6)

Listing 2.5: GET(appWindows)

```
app.get('/appWindows', function(req, res){
  if(isTokenValid(req.body.token)){
      res.type('json');
      res.send(SAGERunningApps);
    } else {
      res.type('text/plain');
      res.statusCode = 401;
      res.send("Error 401: You are not authorized...");
    }
});
```

Listing 2.6: response for GET(appWindows)

```
[
  {
    "id": "application_0",
    "application": "atom_smasher",
    "left": 54,
    "top": 81,
    "width": 4230,
    "height": 4230,
    "icon": "uploads/assets/apps/atom_smasher",
    "title": "Atom Smasher",
    "color": null
  }, ...
]
```

FP4 Pro GET requesty poslední skupinou požadavků je získání seznamu souborů. S ohledem na teprve probíhající vývoj cloudového řešení napojeného na SAGE2 jde zatím o jedinou API funkci poskytující seznam souborů. Seznam je sice rozdělen podle typu souborů, ale jakékoliv filtrování či vyhledávání není ze strany aktuálního správce souborů na SAGE2 podporováno. Další funkce budou tedy realizovány až s hotovým cloudovým řešením.

URL '/files' poskytuje seznam dustupných souborů v následujícím formátu:

Listing 2.7: response for GET(files)

```
{
  "images": [
    {
      "filename": "QR.png",
      "id": "/home/sage/projects/bremondr/public/uploads/images/QR.png",
      "filetype": "PNG"
    }
  ],
  "videos": [
  ],
  "pdfs": [
  ],
  "sessions": [
    {
      "filename": "default"
    }
  ]
}
```

2.1.2 POST

REST requesty POST jsou tradičně využívány k předávání dat sloužících k vytváření nových záznamů v databázi. V případě ovládání systému SAGE2 jsou využity k zapínání aplikací a monitorů a otevírání souborů.

FP1

URL '/monitors' ('monitors/:id') poskytuje přístup k zapínání monitorů po jednom, po skupinách nebo všech najednou. Skupiny monitorů jsou rozpoznány jako záporná čísla, v případě implementace pro telestěnu v SAGE-labu jde o čísla -1 až -5 pro pět sloupců telestěny. Zároveň je implementována kontrola, zda zapnutí požadovaného monitoru je proveditelné, tedy jestli monitor s takovým ID existuje. V případě úspěchu vrací true, v opačném případě chybu "404 Not found". Uvnitř je realizována voláním funkcí *switchOn*, *switchOnGroup* a *runMonitors*, které využívají knihovnu NEC-Controller [2].

FP2

URL '/apps/:id' umožňuje zapnout zvolenou aplikaci na základě ID převzatého z odpovědi na GET request na seznam aplikací. API kontroluje, zda požadované ID existuje. Pokud ano, je proveden požadavek na SAGE2, který spustí aplikaci. V opačném případě je vrácena chyba 404. Jak je vidět z ukázky 2.8, pokud jsou splněny požadavky na autorizaci a platnost ID, je přímo vykonán požadavek na spuštění aplikace.

Listing 2.8: POST(apps/id)

```
app.post('/apps/:id', function(req, res){
  if(isTokenValid(req.body.token)){
        if(isValid(req.params.id, SAGEApps )){
          masterConnectionToSAGE.send(
              JSON.stringify({"f":"loadApplication","d":
                  {"application": req.params.id,
                   "user":SAGEInit.UID}}));
          res.json(true);
        } else {
          res.type('text/plain');
          res.statusCode = 404;
          res.send("Error 404: No such app found");
        }
    } else {
        res.type('text/plain');
        res.statusCode = 401;
        res.send("Error 401: You are not auth...");
    }
});
```

URL '/files/:fileID' otevírá soubor specifikovaný jeho ID v adekvátní aplikaci. Dalším parametrem nutným pro funkčnost je tedy specifikace typu souboru, o který se jedná. Mezivrstva na základě toho zvolí vhodnou aplikaci, ověří platnost ID souboru a autorizovanému uživateli tak umožní otevření požadovaného souboru na telestěně. Vnitřní realizace je pak velmi podobná ukázce 2.8 s tím rozdílem, že je nutné určit, jakou aplikaci zvolit pro otevření požadovaného souboru. Jelikož seznam aplikací neobsahuje informaci o tom, jaké soubory která aplikace podporuje, je výběr vhodné aplikace zatím řešen napevno v kodu. S realizací nového správce souborů by mohla přibýt možnost získání seznamu podporovaných souborů ke každé aplikaci. Typ otevíraného souboru by se pak porovnal s tímto seznamem a podle shody by se zvolila aplikace.

FP3 Skupina požadavků, která se v kategorii GET nevyskytla. Jedná se o ovládání kurzoru, ke kterému žádné informace zpět ze SAGE2 poskytovány nejsou. Zde tedy jen jediný POST request na spuštšní kurzoru.

URL '/pointer' při spuštění kurzoru na SAGE2 předá informaci o popisu kurzoru a také o jeho barvě. Kurzor je vytvořen na SAGE2

2.1.3 PUT

Ekvivalentně ke standartnímu použití REST requestu PUT na úpravy položek, umožňuje nastavovat vlastnosti zapnutých monitorů a také ovládat okna a kurzor na telestěně.

FP1

URL '/monitors/backlight/:value' a '/monitors/:id/backlight/:value' umožňují nastavení podsvícení monitorů telestěny. Jsou tři možnosti využití této části API. Buď lze ovládat celou telestěnu najednou a nebo po jednotlivých monitorech. Doplňující funkcí je pak možnost ovládání skupin monitorů skrytých pod ID '-1' až '-5' pro pět sloupců.

Listing 2.9: PUT(monitors/backlight)

```
app.put('/monitors/backlight/:value', function(req, res){
  if(isTokenValid(req.body.token)){
      setBacklightOnAll(req.params.value);
      res.json(true);
  } else {
      res.type('text/plain');
      res.statusCode = 401;
      res.send("Error 401: You are not authorized...");
    }
});
```

FP2 obsahuje požadavky umožňující úpravy oken aplikací běžících na SAGE2 serveru. Zároveň se jedná o část API, která vyžadovala úpravy na kodu SAGE2 tak, aby bylo možné obsluhovat okna na základě specifikovaného ID. Bez této úpravy by nebylo možné vybrat konkrétní okno a pohnout s ním. Implementace SAGE2 zde původně pohyb oken obstarávala dříve popsanou analýzou pozice a pohybu virtuálního kurzoru. Nevýhody tohoto řešení byly popsány v předešlých částech práce.

URL '/appWindows/:id' s parametry x, y, width a height slouží ke změně velikosti a polohy okna. Souřadnice i rozměry jsou udávány v pixelech odpovídajících pixelům telestěny. V ukázce 2.10 je vidět využití nově nadefinované zprávy "moveWindow" příjmané SAGE2 a jednak ověření platnosti ID.

Listing 2.10: PUT(appWindows/id)

```
app.put('/appWindows/:id', function(req, res){
  if(isTokenValid(req.body.token)){
        var index = findWindowById(req.params.id);
        var app = SAGERunningApps[index];
        if(app != null){
          masterConnectionToSAGE.send(JSON.stringify(
              {"f": "moveWindow", "d":
              {"x": req.body.x, "y": req.body.y,
               "width": req.body.width,
               "height": req.body.height,"id": index} }
```

```
            ));
          res.json(true);
        } else {
          res.statusCode = 404;
          res.send('Error 404: No window with such id');
    } else {
        res.type('text/plain');
        res.statusCode = 401;
        res.send("Error 401: You are not authorized to acces this function");
    }
});
```

URL '/appWindows/:id/zoom/:amount' umožňuje zvětšit nebo zmenšit okno bez jeho přemístění o určité procento specifikované parametrem "amount". Parametr amount je pak brán jako procentuální velikost výsledného okna. To znamená že "amount":150 zvětší okno na 1,5násobek, analogicky "amount":50 dané okno zmenší na polovinu. Po stránce kontroly vstupů a vlastního provedení změn na okně je funkce shodná s předchozí ukázkou, s výjimkou posílané ws zprávy.

FP3 skupina požadavků s URL prefixem **'/pointer/'** obstarává obsluhu kurzoru. Požadavky jsou tří druhů. První skupinou jsou požadavky **leftPress**, **leftRelease** a adekvátní požadavky pro pravé tlačítko myši. Tato skupina slouží v případech kdy je potřeba kurzorem cokoliv uchopit a někam přemístit. Pro přemístění je zde API funkce move mající jako parametry dx a dy což jsou rozdíly proti předchzí poloze. SAGE2 bohužel neposkytuje informace o aktuální poloze kurzoru, a proto je nutné, aby si tuto informaci udržovala klientská aplikace sama.

Druhou skupinou požadavků jsou klikací požadavky **leftClick** a **rightClick**, které by se daly nahradit společným použitím předcházejích funkcí, ovšem takto je snížena zátěž sítě a hlavně eliminována možnost posunu kurzoru mezi press a release požadavkem, která může v určitých případech způsobovat nežádoucí jevy.

Do poslední skupiny patří speciální funkce **doubleClick** eliminující opět možnost nechtěného posunu mezi kliknutími a funkce scroll pro simulování kolečka myši. Jedno **scroll** volání odpovídá jednomu "cvaknutí" kolečkem myši.

2.1.4 DELETE

Poslední skupina REST požadavků je v souladu s konvencemi použitá k odstraňování elementů. V případě telestěny to znamená zavírání běžících oken, vypínání monitorů a odstranění kurzoru.

URL '/wall' slouží k ukončení činnosti na telestěně. Provede zavření oken všech aplikací a zároveň vypne všechny monitory.

Listing 2.11: DELETE(wall)

```
app.delete('/wall', function(req, res){
  if(isTokenValid(req.body.token)){
        for(var i = 0; i < SAGERunningApps.length; i++)
          masterConnectionToSAGE.send(JSON.stringify({"f":"killWindow",
             "d":{"id": SAGERunningApps[i].id,"user":SAGEInit.UID}}));
        killMonitors();
        res.json(true);
    } else {
        res.type('text/plain');
        res.statusCode = 401;
        res.send("Error 401: You are not authorized to acces this function");
    }
});
```

URL '/monitors' a /monitors/:id zajišťuje vypínání monitorů jednotlivě, po skupinách nebo najednou. Určení skupin je stejné jako v případě zapínání a úpravy podsvícení.

URL '/appWindows' a '/appWindows/:id' vypne všechny nebo specifikovanou aplikaci.

URL '/pointer' odstraní kurzor myši z telestěny.

2.2 WebSocket

Realizace websocketové komunikace byla provedena využitím knihovny ws. Po důkladném pozorování a shromažďování zpráv posílaných mezi SAGE2 serverem a implmentovaným webovým UI byl získán ucelený seznam a přehled o komunikaci, kterou SAGE2 vyžaduje. Formát zpráv posílaných SAGE2 je popsán v kapitole *"API poskytované SW SAGE2"* a seznam zpráv je k prostudování v příloze *"Websocketové zprávy systému SAGE2"*. Pro příjem zpráv bylo nutné vytvořit jedno hlavní spojení se serverem a zprávy obdržené v rámci tohoto spojení zpracovat a připravit pro klienty, kteří se na jejich obsah dotazují přes implementované REST API. Navázání spojení je vidět na následující ukázce. 2.12.

Listing 2.12: Navázání hlavního ws spojení se SAGE2

```
var masterConnectionToSAGE = new ws('ws:195.113.232.57:8484');
masterConnectionToSAGE.on('open', function open() {
    masterConnectionToSAGE.send(JSON.stringify(masterInitMessage));
    masterConnectionToSAGE.send(JSON.stringify(
          {"f":"requestAvailableApplications"}));
  });
```

Komunikace přicházející z SAGE2 serveru je zpracována v “event listeneru” pro příchozí zprávy (ukázka č. 2.13). Vzheldem k formátu ws zpráv, který obsahuje název zprávy a data zprávy, jsou všechny příchozí zprávy roztříděny ke zpracování podle svého jména. Z toho důvodu není nutné analyzovat složitě data, která zpráva obsahuje.

Listing 2.13: Listener funkce pro příchozí zprávy ze SAGE2

```
masterConnectionToSAGE.on('message', function(data, flags) {
      switch(JSON.parse(data).f){
        case 'initialize':
           SAGEInit = JSON.parse(data).d;
           break;
        case 'availableApplications':
           getApps(JSON.parse(data).d)
           break;
        case 'setupDisplayConfiguration':
           SAGEConfiguration = {
             ....
           };
           break;
        case 'createAppWindowPositionSizeOnly':
           SAGERunningApps.push(JSON.parse(data).d);
           wss.broadcast(SAGERunningApps);
           break;
        case 'setItemPosition':
          ....
        case 'setItemPositionAndSize':
           for(var i = 0; i < SAGERunningApps.length; i++){
              if(SAGERunningApps[i].id == JSON.parse(data).d.elemId){
                SAGERunningApps[i].left   = JSON.parse(data).d.elemLeft;
                SAGERunningApps[i].top    = JSON.parse(data).d.elemTop;
                SAGERunningApps[i].width  = JSON.parse(data).d.elemWidth;
                SAGERunningApps[i].height= JSON.parse(data).d.elemHeight;
              }
           };
           break;
        case 'deleteElement':
          ....
           break;
        case 'updateItemOrder':
          ....
          break;
        case 'storedFileList':
          ....
        default:
          console.log("default action "+JSON.parse(data).f);
          console.dir(JSON.parse(data).d);
          break;
      }
      wss.broadcast(JSON.stringify(SAGERunningApps));
  });

  });
```

Takto identifikované zprávy jsou pak na základě znalosti formátu dat, která obsahují, zpracovány a data přímo uložena nebo dále zpracována ještě před uložením. Zpracovávají se data posílaná v seznamu aplikací a souborů, jelikož pro klientské aplikace je podstatných velmi málo informací. Zprávy posílané SAGE2 obsahují velké množství nadbytečných informací, proto jsou v rámci předzpracování ze zpráv vybrány jen informace relevantní pro použití na mobilních aplikacích. Snižuje se tím množství přenášených dat i náročnost parsování na straně klientských UI aplikací. Snížení množství přenášených dat

je způsobeno tím, že v mobilní UI aplikaci není potřeba znát reálné umístění souboru a všechny jeho vlastnosti. Pro otevření souboru stačí znát jeho název a o jaký typ se jedná. V ukázce č. 2.14 je vidět příklad zpracování seznamu dostupných aplikací. Z dostupných dat posílaných ze strany SAGE2 je pro klientskou aplikaci užitečný pouze název aplikace, který slouží jako identifikátor k jejímu spuštění, a zároveň slouží také uživateli. Do budoucna by bylo vhodné přidat stručný popis toho, co aplikace dělá, a jaké podporuje formáty otevíraných souborů. Tyto informace ovšem zatím nejsou v datech poskytovaných SAGE2 dostupné a není je tak možné využít.

Listing 2.14: Funkce zpracovávající seznam aplikací

```
function getApps(data){
  data.forEach(function each(app) {
    SAGEApps.push({"name":app.exif.FileName});
  });
}
```

Když jsou příchozí zprávy takto zpracovány a uloženy do paměti pro pozdější využití, je klientům pomocí ws rozšíření API rozeslána zpráva s aktualizací stavu telestěny (závěr ukázky č. 2.13). Jak již bylo řečeno, pokud klient nepodporuje příjem těchto ws aktualizačních zpráv, je aktualizace ponechána na jeho zodpovědnost pomocí volání patřičných REST dotazů.

2.3 Vnitřní realizace

Rozhraní API je podporováno sadou interních funkcí, jejichž kompletní popis je součástí dokumentace práce. Jedná se především o funkce ověřující platnost parametrů požadavků a ověření oprávnění uživatele. Ověření uživatele je přitom pouze připravené na připojení shibboleth, které bude realizováno, až bude připravené API aplikace shibboleth. Z ukázek níže je pak vidět část realizující připojení monitorů pomocí NEC-Controller knihovny. Každý z monitorů komunikuje na vlastní IP adrese, přes kterou je do Controlleru připojen. Další ovládání monitoru je pak realizováno přes ID zadané každému monitoru při inicializaci spojení.

Listing 2.15: Inicializace NEC-Controlleru

```
function initMonitorsControler(){
  console.log("inivializing monitors controller");
  faistapat.initController();
  for(var i = 0; i < 20; i++){
    if(i<10)
      faistapat.addMonitor('10.0.0.10'+i, 7142, i+1);
    else
      faistapat.addMonitor('10.0.0.1'+i, 7142, i+1);
  }
  faistapat.connectAll();
}
```

S ohledem na problémy knihovny ffi sloužící k importu C++ knihovny funkcí, způsobující nemožnost odchytávat vyjimky vyhozené při volání C++

funkcí, je nutné ošetřit vstupy a volání těchto funkcí tak, aby k vyhození vyjimek nedocházelo. Další možnou variantou by bylo přizpůsobení knihovny NEC-Controller tak aby místo vyhazování vyjimek vracela návratové kody funkcí určující jaká chyba nastala. Tato možnost by vyžadovala přepsání NEC-Controlleru a proto byla ponechána jako alternativa pro budoucí vývoj.

2.4 Úpravy provedené na SAGE2

S ohledem na výsledky analýzy, které prokázaly nepraktičnost některých websocketových volání, byly v rámci této práce doplněny některé funkce přímo do SW SAGE2. Jedná se o přidání listener funkcí, které zpracovávají příchozí websockety a dále funkcí vykonávajících potřebné operace na základě požadavků. Především se jedná o nahrazení ovládání aplikací prostým přenosem pozice kurzoru napojením přímo na vhodné rozhraní aplikací. V rámci práce nejsou realizovány všechny změny, které by se do budoucna hodily, ale jen ty, které byly nezbytně nutné pro zprovoznění mezivrstvy, aby jí poskytované rozhraní bylo vhodné pro mobilní aplikace. Další úpravy by byly nutné zejména s realizací další požadavků na ovládání konkrétních aplikací. Zamýšlené úpravy na, které se nedostalo s realizací v rámci této práce jsou zmíněny v kapitole *"Nerealizované požadavky"*. Konkrétní změny jsou popsány na následující ukázce programu.

V první řadě bylo nutné přidat následující listener funkce, které očekávají ws zprávy a vyvolávají funkce, které je mají zpracovat. V tomto případě se tedy jedná o přidání možnosti zavřít okno aplikace a posunout okno aplikace na základě jeho ID. Dosavadní řešení bylo realizováno pomocí přnosu pozice kurzoru a následné analýzy kliknutí až na serveru SAGE2. Jak je patrné z analýzy UI na mobilních zařízeních, není toto příliš praktické řešení.

Listing 2.16: Přidané event listenery na straně SAGE2

```
wsio.on('killWindow', wsKillWindow);
wsio.on('moveWindow', wsAppMove);
```

Zmíněné listenery pak volají konkrétní implementaci funkcí operujících s okny na základě jejich dodaného ID. Bylo tedy nutné zajistit vyhledání okna podle jeho identifikace namísto polohy kurzoru. Ukázalo se, že patřičná funkce findAppById(id) ve zdrojovém kodu existuje, jen není využita. S využitím této funkce byla tedy implementována funkce na zavírání okna a následně i funkce zajišťující pohyb a změny velikosti daného okna. (viz ukázky 2.17 a 2.18)

Listing 2.17: Přidané funkce pro zavření okna

```
function wsKillWindow(wsio, data){
  var app = findAppById(data.id);
  deleteApplication(app);
}
```

Listing 2.18: Přidaná funkce pro pohyb oknem podle ID

```
function wsAppMove(wsio, data) {
 var app = findAppById(data.id);
 if (app) {
  app.left  = data.x;
  app.top = data.y;
    app.width = data.width;
    app.height = data.height;
  var updateItem = {elemId: app.id,
    elemLeft: app.left, elemTop: app.top,
    elemWidth: app.width, elemHeight: app.height,
    force: true, date: new Date()};
  broadcast('setItemPositionAndSize',
      updateItem, 'receivesWindowModification');
 }
}
```

Všechny změny provedené na SW SAGE2 jsou součástí práce jako patch pro instalaci SAGE2.

2.5 Nerealizované požadavky

S ohledem na průběh vývoje a analýzu softwaru SAGE2 se bohužel nepodařilo realizovat všechny funkční požadavky. Nerealizované požadavky jsou následující:

Napojení aplikací přes API tak, aby bylo umožněno jejich přímé ovládání pomocí tabletu. Jak je patrné z analýzy SAGE2, tak v současné době je jejich ovládání realizováno vyhodnocováním akcí kurzoru. Jak již bylo zmíněno, tato forma se pro práci z mobilního zařízení nehodí, především s ohledem na nepoměr mezi pracovní plochou na tabletu a plochou telestěny. Vzhledem k absenci přímého rozhraní pro napojení aplikací, tak aby jejich ovládací prvky byly na mobilním zařízení a až výsledná akce se přenášela na SAGE2, bylo nutné prozatím tento požadavek opustit. I když jeho část byla realizována úpravami na SAGE2 (viz kapitola *"Úpravy provedené na SAGE2"*), plná realizace by vyžadovala příliš velký zásah do zdrojových kodů SAGE2, který nebyl předmětem zadání, a především vzhledem k častým změnám ze strany vývojářské skupiny SW SAGE2 není rozumně realizovatelný. Návrh aplikace podporuje tuto funkčnost do budoucna, ale implementace byla, po konzultaci s vedoucím práce i ostatními zainteresovanými členy týmu rozvíjejícím SW vybavení laboratoře, odložena do doby, kdy bude k dispozici stabilnější, a snad i lépe dokumentovaná verze SW SAGE2.

Kapitola 3

Testování

Na závěr implmentační části práce bylo zařazeno testování funkčnosti celé aplikace a testování provázanosti s ostatními dílčími částmi projektu SAGE-lab. Testy byly provedeny ve třech kategoriích. Již v průběhu implementace bylo testováno propojení každého REST API požadavku s mobilní aplikací platformy Android. Nalezené nesrovnalosti se, pokud to bylo možné, ihned projevovaly do implementace. Dále probíhalo testování funkcí poskytovaných knihovnou NEC-Controller a API SAGE2. Na závěr byl pro celkové otestování výsledku použit bash script testující všechny části API na různé vstupy. Shodnocení výsledků jednotlivých testů je rozděleno následovně.

3.1 Uživatelské testování z Android klienta

Při testování z Android klienta byly objeveny především nedostatky při práci se vzdáleným kurzorem. Některé jsou způsobené nevhodným chováním systému SAGE2, například funkce napojené na pravé tlačítko jsou těžko využitelné, jiné pak byly způsobené nevhodností technologie REST pro práci s kurzorem. Chyby objevené v průběhu testování byly opraveny a v aktuální verzi by se již neměly vyskytovat. Jedná se například o špatný formát odpovědí či nefungující pohyb s okny aplikací. Podrobnější popis testování je součástí BP *"Android manager pro SAGE"* [1].

3.2 Testy API využívaného mezivrstvou

V průbhu automatického testování byla objevena chyba v NEC-Controlleru, která způsobovala pád aplikace při zadání některých hodnot backlight. Tato chyba byla identifikována a opravena, více v příslušné BP [2]. Testy také odhalily, že SAGE2 nekontroluje ID aplikací, které mají být spuštěné a prostě se pokusí spustit aplikaci s ID, které je mu dodáno. Pokud taková aplikace

neexistuje, je následkem pád aplikace SAGE2. Z toho vyplývá nutnost tuto kontrolu doimplementovat do mezivrstvy.

3.3 Závěrečné automatické testování

Na základě zkušeností z průběžného testování byla speciální pozornost věnována otestování funkčnosti části API pracující s NEC-Controllerem, aby se vyloučil výskyt dalších chyb způsobujících pád aplikace. Testování již neodhalilo další chyby a ovládání telestěny tak funguje spolehlivě.

S ohledem na průběžné testy, které odhalily skutečnost, že SAGE2 nekontroluje platnost ID aplikace, kterou má spustit a v případě neplatného ID spadne, byl implementován kontrolní mechanismus v mezivrstvě. V průběu testování spouštění různaých aplikací, i neexistujících, pak již k pádu systému nedošlo. Testování práce s aplikacemi pak potvrdilo také funkčnost zavírání oken přes API a možnost jejich přesunu a změnu velikosti. V souvislosti s pohybem oken po telestěně byla objevena anomálie, při které se okno přesune jak má, na telestěně je také zobrazené v pořádku, ale na náhledu ve webovém UI aplikace SAGE2 je zminimalizováno. Po kliknutí na zminimalizované okno v tomto UI se obnoví do správných rozměrů. Bohužel se nepodařilo najít příčinu tohoto chování.

Při práci se soubory byla v průběhu testování odhalena chyba spočívající v nemožnosti otevřít jakýkoliv soubor dříve, než se klientská aplikace dotáže na seznam souborů. I když je nepravděpodobné, že by tato situace v praxi nastala, byla provedena náprava a v současné verzi již nezáleží na tom, zda se klient dotáže na seznam souborů nebo ne.

3.4 Shrnutí

K testování byl použit bash script využívající nástroje *"curl"*[**?**] a doplněk do prohlížeče Chrome *"REST Console"*[**?**]. Testování odhalilo malé chyby způsobené na úrovni vlastního programování API, které byly opraveny. Zásadnější nedostatky odhalené při testování způsobené využitými technologiemi a problémy se systémem SAGE2 byly zaznamenány a v kapitole *"Zhodnocení"* je navrženo jejich řešení v budoucích verzích API.

Kapitola 4

Zhodnocení

Na základě výsledků testování, analýzy a komplikací objevených při implementaci a napojování na související aplikace bylo objeveno několik oblastí práce, které by bylo vhodné do budoucna upravit. Obecně se jedná především o problémy se samotným systémem SAGE2, který obsahuje mnoho úskalí a nelogických postupů. Klíčové oblasti vyžadující další analýzu a hledání řešení jsou tyto:

- SAGE2 využívá při sdílení kurzoru pravé tlačítko myši, což na jednu stranu dává smysl, ovšem při napojování mobilních zařízení to přináší jisté problémy, které bude třeba dále, ve spolupráci s koncovými uživateli, řešit.
- REST technologie nepodporuje jednostranné vyvolání komunikace ze strany serveru. To je problém hlavně z důvodu, že SAGE2 je zamýšlen jako kooperativní platforma, ve které najednou pracuje na jedné telestěně více lidí. Tento nedostatek technologie byl vyřešen přidáním websocketového API do mezivrstvy tak, aby mohly být aktualizace stavu telestěny posílány zpět klientům kdykoliv.
- Stejná vlastnost RESTu způsobuje problémy i se vzdáleným sdílením kurzoru. V tomto případě se problémy objevily až ve fázi testování. V dalších verzích aplikační mezivrstvy by tak bylo vhodné upravit API tak, aby se i ovládání kurzoru přesunulo na technologii websocket, což umožní snazší svázání klienta se svým kurzorem.
- SAGE2 řeší ovládání aplikací formou analýzy pozice kurzoru, což se pro mobilní aplikace ukázalo jako nevhodný postup. Přímé napojení na ovládací prvky je pak plánováno do budoucna, až bude k dispozici stabilnější verze SAGE2. Zároveň je nutné provést komplexní analýzu zdrojových kodů systému SAGE2 a pokud možno je zdokumentovat.

Závěr

Práce se zabývala analýzou systému SAGE2 pro telestěny zejména s pohledu UI a napojení mobilních zařízení k ovládání telestěny. Ananlýza odhalila nedostatky v uživatelském rozhraní způsobené z části zvolenou technologií provedení a zčásti neresponzivní realizací webového UI. Bylo zjištěno, že ovládání virutálním kurzorem je pro mobilní zařízení nevhodné. Dále byly analyzovány požadavky mobilní aplikace na rozhraní nutné pro vytvoření použitelného UI.

Na základě analýzy současného stavu a dostupných technologií bylo navrženo REST API jako univerzální rozhraní pro ovládací aplikace. Toto REST API zpřístupňuje ovládání mobilním aplikacím a umožňuje výrazné úpravy UI bez ohledu na vývoj probíhající na sw SAGE2.

Návrh byl následně implementován technologií NodeJS. Realizace navrženého API si v průběhu práce vyžádala nečekané zásahy přímo do zdrojů SAGE2, tak aby bylo možné splnit požadavky kladené mobilní aplikací.

Realizované API bylo následně otestováno ve spolupráci s ostatními členy SAGElab projektu. Byla ověřena funkčnost jednotlivých částí rozhraní i provázanost s mobilní aplikací na Androidu. Odhalené nedostatky a chyby byly pokud možno odstraněny. V případě komplikací souvisejících s použitou technologií byl vytvořen návrh řešení do budoucna.

Otestované API splňuje až na drobné výjimky popsané v příslušné kapitole všechny požadavky vzešlé z analýzy. Implementace autentifikace pomocí technologie shibboleth nebyla plně implementována jelikož nebylo připraveno rozhraní, které se mělo využít. Autetifikace je tak pouze připravena na realizaci v další verzi API.

Do dalších verzí bude dále vhodné implementovat podporu ovládání jednotlivých aplikací přímo z mobilní aplikace bez nutnosti sdílet kurzor. V blízké době bude upraveno napojení správce souborů na vyvýjené cloudové řešení namísto aktuálního správce souborů integrovaného v SW SAGE2. V připravovaném dalším vývoji je také počítáno s analýzou a realizací ovládání audio výstupů telestěny. Tím by pro uživatele bylo dosaženo integrace všech ovládacích prvků do jednoho rozhraní a práce se stěnou by se tak výrazně usnadnila.

Literatura

[1] Pham Tat, Dat. SAGELab - Android Manager zařízení SAGE. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

[2] Faistaver, Patrik. Implementace knihovny a aplikace pro ovládání LCD monitorů NEC X463UN. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

[3] Kubišta, Jiří. SAGELab - Distribuované zobrazování obrazových dat ve vyso-kém rozlišení na zařízení SAGE. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015

[4] ExpressJS [online]. [cit. 2015-05-05]. Dostupné z: http://expressjs.com

[5] NodeJS. 2015. NodeJS [online]. [cit. 2015-04-07]. Dostupné z: https://nodejs.org/

[6] ROTTMANN, Peter. 2015. ApiDoc [online]. [cit. 2015-05-06]. Dostupné z: http://apidocjs.com/

[7] STANGVIK, Einar Otto. 2015. Ws: a node.js websocket library [online]. [cit. 2015-05-06]. Dostupné z: https://github.com/websockets/ws

[8] SAGELAB,. 2015. SAGELab: Síťová multimediální laboratoř [online]. [cit. 2015-05-06]. Dostupné z: http://sagelab.cesnet.cz/o-laboratori/

[9] LAVA,. 2015. SAGE2: Scalable Amplified Group Environment [online]. [cit. 2015-03-06]. Dostupné z: http://sage2.sagecommons.org/

[10] WebSocket Demos. 2013. WebSocket.org [online]. [cit. 2015-05-06]. Dostupné z: https://www.websocket.org/demos.html

[11] NPM. 2015. Npm body-parser [online]. [cit. 2015-05-06]. Dostupné z: https://www.npmjs.com/package/body-parser

[12] NODE.JS AND EXPRESS: CREATING A REST API. 2015. Modulus [online]. [cit. 2015-04-06]. Dostupné z: http://blog.modulus.io/nodejs-and-express-create-rest-api

[13] Build a RESTful API Using Node and Express 4. 2015. Scotch.io [online]. [cit. 2015-04-06]. Dostupné z: https://scotch.io/tutorials/build-a-restful-api-using-node-and-express-4

[14] Writing WebSocket client applications. 2015. Mozila developer network [online]. [cit. 2015-04-06]. Dostupné z: https://developer.mozilla.org/en/docs/WebSockets/Writing_WebSocket_client_applications

[15] SAGE2 Wiki. 2015. SAGE2 [online]. [cit. 2015-04-06]. Dostupné z: https://bitbucket.org/sage2/sage2/wiki/Home

[16] Learn REST: A RESTful Tutorial. 2015. REST API tutorial [online]. [cit. 2015-04-06]. Dostupné z: http://www.restapitutorial.com/

[17] Introducing WebSockets: Bringing Sockets to the Web. 2010. Html5rocks [online]. [cit. 2015-04-06]. Dostupné z: http://www.html5rocks.com/en/tutorials/websockets/basics/

[18] SAGE2(tm) Software License Agreement. 2015. SAGE2 [online]. [cit. 2015-05-06]. Dostupné z: https://bitbucket.org/sage2/sage2/src/0aae98c3ca879fa215610e27847ed2cc96ea8343/LICENSE.txt?at=master

[19] SAGE2 / Install. 2015. SAGE2 [online]. [cit. 2015-05-06]. Dostupné z: https://bitbucket.org/sage2/sage2/wiki/Install%20(Ubuntu)

Příloha A

Seznam použitých zkratek a pojmů

GUI Graphical user interface

UI User interface (uživatelské rozhraní)

telestěna 20 monitorů využitých pomocí sw SAGE jako jeden "monitor"

SAGE Scalable Adaptive Graphics Environment v práci užíváno jako SAGE1

SAGE2 Scalable Amplified Group Environment

API Application Programming Interface

DB Databáze

WS WebSocket

SW Software

HW Hardware

BP Bakalářská práce

FP Funkční požadavky

NP Nefunkční požadavky

OS Operační systém

UHD Ultra High Definition (Ultra vysoké rozlišení)

JSON JavaScript Object Notation

SAGElab Multimediální a síťová laboratoř na FIT ČVUT

Příloha **B**

Instalační příručka

Pro zprovoznění mezivrstvy na vlastním počítači je nutné mít zprovozněný vlastní server. Na tomto serveru pak musí být zporovzněny závislositi požadované SW SAGE2.

1. Nainstalovat SAGE2 podle příručky dostupné zde: [19]
2. Aplikovat patch ze souboru SGAE2patch na soubor server.js z instalace SAGE2
3. Nainstalovat knihovnu express.js pomocí příkazu *npm install express –save*
4. Nainstalovat knihovnu body-parser npm pomocí příkazu *npm install body-parser*
5. Nainstalovat knihovnu ws pomocí příkazu *npm install ws*
6. Nainstalovat knihovnu ffi pomocí příkazu *npm install node-ffi*
7. Zkopírovat složku src do umístění odkud se bude spouštět
8. Do souboru *api-layer* upravit adresu a port vlastního SAGE2 serveru
9. Spustit soubor *api-layer* ve složce src

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `bremondr_app` ..... spouštěcí skript
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `SAGE2patch` ..... patch pro SAGE2
- `docu`
  - `apiDocu` ..... Dokumentace API
  - `srcDocu` ..... Dokumentace
- `text` ..... text práce
  - `BI-bremondr.pdf` ..... text práce ve formátu PDF
  - `messages.pdf` ..... WebSocketové zprávy systému SAGE2
  - `BI-bremondr` ..... zdrojová forma práce ve formátu LaTeX