Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Systém pro distibuci a přehrávání multimediálního obsahu

Bc. David Krulich

Diplomová práce
2016

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2015/2016**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. David Krulich**
Osobní číslo:	**I14266**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Systém pro distribuci a zobrazování multimediálního obsahu**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem práce je navržení vlastního systému pro distribuci a zobrazování multimediálního obsahu na klientech.
Klientem je myšleno zařízení, které je schopno zobrazit multimediální obsah a je připojeno do distribuční sítě. Klient je schopen ze serveru získávat buď hotový multimediální obsah nebo scénář s příslušnými zdroji.
Součástí teoretické části bude provedena rešerše na téma existujících systémů automatizovaného šíření multimediálního obsahu. Dále budou v teoretické části popsány vhodné technologie pro komunikaci v síti klient-server a technologie pro vytváření multimediálních aplikací.
V rámci praktické části bude provedena kompletní analýza, návrh a implementace požadovaného systému pro distribuci a zobrazování multimediálního obsahu. Systém bude pokrývat celý proces od vložení jednotlivých zdrojů do systému, přes jejich případné zpracování dle scénáře až po samotnou distribuci a koncovou reprodukci.
Systém by měl být dostatečně variabilní z pohledu způsobu a místa zpracování scénáře a podkladů, z pohledu způsobu distribuce atd. Součásti řešení je i tvorba nástroje pro editaci scénáře a nástroje pro renderování finálního multimediálního obsahu dle scénáře. Další součástí systému je vlastní multimediální přehrávač pro zobrazování multimediálního obsahu dle zadaného scénáře.

Rozsah grafických prací:

Rozsah pracovní zprávy: **cca 60 stran**

Forma zpracování diplomové práce: **tištěná**

Seznam odborné literatury:

ARLOW, Jim a Ila NEUSTADT. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. Brno : Computer Press, 2007, 567 s. ISBN 978-80-251-1503-9.
PETZOLD, Charles. Mistrovství ve Windows Presentation Foundation. Brno: Computer Press, 2008, 928 s. Mistrovství. ISBN 978-80-251-2141-2.

Vedoucí diplomové práce: **Ing. Petr Veselý**
Katedra softwarových technologií

Datum zadání diplomové práce: **31. října 2015**
Termín odevzdání diplomové práce: **13. května 2016**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2015

Prohlašuji:
Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.
Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 10. května 2016

Bc. David Krulich

Děkuji vedoucímu práce Ing. Petru Veselému
za neocenitelné rady a pomoc při tvorbě diplomové práce.

ANOTACE

V této diplomové práci jsem se zaměřil na návrh a vývoj aplikace zprostředkující distribuci a následné přehrávání multimediálního obsahu – aplikace se celo-světově nazývají *Digital Signage*. Na základě rešerše již existujících systémů byl navržen systém vlastní. K návrhu dopomáhala metodika UML. Implementace byla provedena formou webové aplikace pod technologií HTML5. Implementováno bylo vykreslovací jádro vytvářející plynulý obraz na základě předdefinovaného scénáře, grafický editor scénáře založený na čase a síťová komunikace prvků systému pracující v reálném čase.

KLÍČOVÁ SLOVA

distribuce digitálního obsahu, renderování videa, real-time síťová komunikace, Node.js, JavaScript

ANNOTATION

In this master thesis I focused on the design and development of the aplication mediating distribution and subsequent playback of multimedia content - worldwide called Digital Signage. The own system was designed on the basis of research of the existing systems. UML methodology was used for the design and implementation was performed using web aplication under HTML5 technology. Rendering engine producing smooth frame rate based on predefined scenario, was implemented, as well as grafical editor of scenarios based on time and network communication of the elements working in a real time.

KEYWORDS

digital content distribution, video rendering, a real-time network communication, Node.js, JavaScript

Obsah

Seznam obrázků

Seznam tabulek

Terminologie

.NET: je technologie od firmy Microsoft určená pro vývoj software
API: Application Programming Interface - sbírka procedur, funkcí, tříd a protokolů určité knihovny
Cookie: rozšiřuje bezkontextový protokol http o možnost uložení konkrétních dat na klientský počítač – jedná se o textová data
Cache: je vyrovnávací paměť
Callback: je funkce, která je provolaná v případě dokončení určité operace. Nahrazuje synchronní návratové klíčové slovo *return*
Color picker: Označení komponenty pro výběr barvy
Controller: je v kontextu vývoje software třída zapouzdřující funkčnost
CRT: jedná se o typ zobrazovacího zařízení – obrazovky
Časová osa: linka založená na čase, jejíž kterýkoliv bod ležící na lince vyjadřuje časový okamžik
Dashboard: je obrazovka, která sdružuje více různých informací. Bývá informativního charakteru, doplněná o grafy apod.
Digital signage: světově užívaný termín pro označení systémů zabývajících se šířením digitálního obsahu
Divák: člověk konzumující audio-vizuální dílo skrze digital signage
Drop down list: je grafická komponenta pro výběr hodnot z číselníku
Framework: je softwarová struktura určující určitý základ pro další aplikace
Full HD: udává rozlišení obrazovky – 1920x1080 pixelů
GUI: Graphical User Interface – uživatelské rozhraní
HTML DOM: Document Object Model – popisuje strukturu HTML dokumentu
Internet of things: neboli internet věcí, určuje jakým způsobem mají být propojena a jak mají komunikovat různá zařízení
ISO/OSI: je referenční model popisující standardizaci počítačových sítí
Multimediální obsah: obsah kombinující alespoň tři složky – např.: audio, obraz a text
Mapování na časovou osu: bodu ležícímu na lince se přidělí určitá událost
Machine-to-machine: v informatice se jedná o komunikační protokol určující způsob komunikace mezi dvěmi elektronickými zařízeními
Lambda výraz: je anonymní funkce, která může být předána jako parametr funkce či vrácena jako návratová hodnota
LCD: jedná se o typ zobrazovacího zařízení – obrazovky

Path rewriting: v rámci webových aplikací se využívá k maskování reálných cest k souborům – navenek v rámci url je uvedena jiná cesta, než je skutečnost v souborovém systému
Plasmový monitor: jedná se o typ zobrazovacího zařízení – obrazovky
Projektor: jedná se o typ zobrazovacího zařízení – slouží k promítání obrazu
Raspberry PI: je jednodeskový počítač o velikosti platební karty
REST API: je webová architektura rozhraní definující přístup k datům postavená na protokolu http
Session: rozšiřuje bezkontextový protokol http o možnost uložení konkrétních dat spjatých s konkrétním klientem
Síťový socket: tato technologie umožňuje procesu pojmout jiný proces běžící na jiném počítači jako svého komunikačního partnera – pomocí počítačové sítě
SOAP: Simple Object Access Protocol - protokol popisující výměnu dat skrz počítačovou síť pomocí textového formátu XML
Streaming: přehrávání série snímků za podpory audia, kdy není celý video–soubor k dispozici, stažená informace je ihned zobrazena
TCP/IP: v informatice se jedná o síťový protokol
Timeline: v kontextu práce s videem se jedná o časovou osu – mapování audio-vizuálních souborů na čas
View: je v kontextu vývoje software třída zapouzdřující grafický interface aplikace

Úvod

Cílem této diplomové práce je uvést čtenáře do problematiky digitálních zobrazovacích zařízení určených pro použití ve veřejných prostorách. Jedná se o zařízení, se kterými se čtenář může nejčasteji setkat v každodenním životě např. v nákupních centrech, v úředních budovách, na náměstích a dalších veřejných místech. Takováto místa jsou právě nejčastěji vybavována přístroji popsanými v této práci. Samotná zařízení jsou určena k šíření informací široké veřejnosti. Může se jednat například o zpravodajství, navigaci v okolí umístění přístroje či marketingové účely.

V této práci budou sjednoceny různé pohledy a požadavky na popisovaný systém do jednotné kategorizace. Na základě navržených kategorií bude vytvořena specifikace jednotlivých kategorií s popisem částí, kterých se daná kategorie týká. Dále budou v této práci navrženy přímé i nepřímé požadavky na systém s vlastnostmi vykazujícími podobné chování jako má systém popisovaný. Zahrnuté budou definice možných přístupů k distribuci informací i jejich následná audio-vizuální reprodukce. Další část práce bude pojednávat o aktuálně nabízených komerčních i nekomerčních systémech. Na základě těchto cizích řešení a vlastního úsudku bude navrženo řešení vlastní. Vlastní řešení bude navrženo s využitím UML metodiky a půjde o definování požadavků, případů užití, analytických diagramů, diagramů tříd atd. Následně bude navržený systém implementován a vybrané zajímavé části budou popsány v této práci. Součástí práce bude i navržení určitých programových testů pro posouzení souladu implementace s návrhem případu užití.

Pro zde řešenou problematiku se globálně rozšířilo slovní spojení *digital signage*. Z důvodu velkého rozšíření tohoto označení si dovolím toto slovní spojení převzít a použít ve své práci i přesto, že se nejedná o český termín.

Má motivace pro zpracování tohoto tématu je z velké části osobní, protože mě samotného zajímají principy, kterými je dosaženo výsledných efektů. Jde o to zajistit, aby správa šířených informací v systému neznamenala vytisknout velké množství plakátů a následně se fyzicky přesouvat mezi jednotlivými cílovými místy pro jejich zveřejnění. Minoritní částí motivace je i fakt, že jsem měl možnost osobně spolupracovat v komerčním světě s podobným systémem, který je ovšem napsán velmi jednoúčelově. Proto je pro mě výzvou toto uchopit dle mých znalostí a dovedností a navrhnout systém, který bude mít vyšší ambice.

1 Teoretické aspekty Digital Signage

1.1 O Digital Signage

Pod označením *signage* si čtenář může představit ceduli, plakát či reklamní tabuli (obr. 1.1). Tato cedule na sobě nese určitou neměnnou informaci. V silniční dopravě to může být např. upozornění na prudkou zatáčku, často se tvořící kolony či vzdálenost k nejbližšímu velkému městu. Cedule umístěná před obchodním domem může informovat kolemjdoucí, že obchodní dům vypsal nové výběrové řízení na post vedoucího určitého oddělení a exkluzivně dneškem rozšířili nabídku zboží o nové produkty.

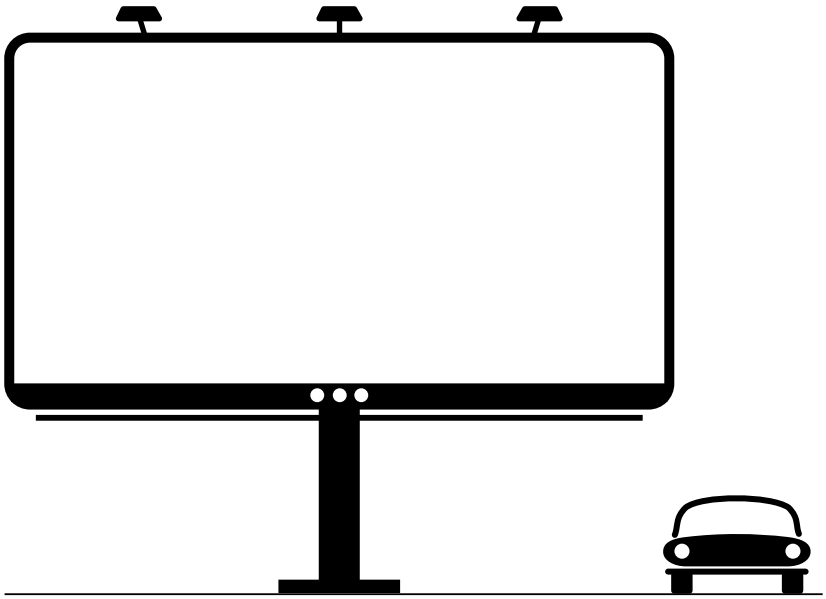
Vysvětlení, co v tomto pojetí znamená slovo *digital*, se dá vyvodit pomocí následujících příkladů. V rámci stále se rozvíjející technologie však může dojít velmi často k požadavku na změnu informačních prvků např. v dopravě. V takovém případě se stane napevno připevněná cedule se statickým obsahem nedostačujícím prvkem a je nutno ji nahradit. Ideálním nahrazením je digitální prvek, který je schopen dynamicky reagovat na aktuální situaci. Tím je právě *digital signage*.

Výhody, které přináší digitálnost, lze vidět na následujících příkladech. Pokud se na dálnici vyskytne kolona, je dobré o ní řidiče informovat. V případě statického řešení je možné na místo častého výskytu kolon dát pevnou značku upozorňující na časté kolony, ale často se kolony vytvoří v návaznosti na nehodu, kterou nelze předvídat. To by se dalo vyřešit tím, že by se ke každé koloně umísťovala přenosná značka, ale to není příliš praktické. Ideálním řešením jsou digitální tabule rozmístěné rovnoměrně po celé dálnici. Ty mohou nejen upozorňovat na aktuální kolonu, ale také omezit rychlost a upozornění doplnit informacemi o vzdálenosti ke koloně, její délce nebo časové prodlevě v dojezdu k nejbližšímu velkému městu.

Smysl digitálnosti lze najít také např. v marketingu. Manager obchodního domu chce reklamní tabulí před obchodem přilákat více kolemjdoucích na exklusivní rozšíření jeho produktové nabídky. Dojde k závěru, že nápis nestačí, že potřebuje na reklamní tabuli zobrazit také fotografie maximálního počtu nového zboží se stručným popisem každého produktu. V případě většího množství zboží by však byla potřeba velká plocha, aby se na ni vše vešlo. Kolemjdoucí navíc nemají chuť trávit minuty prohlížením velkého plakátu, který vlastně ani nechtěli vidět. Proto je v takovém případě papírová cedule opět nedostačujícím prkvem. Jako řešení se hodí systém, který dokáže v určitém intervalu měnit svůj obsah. Čtenář může

namítnout, že pokud se bude obsah měnit, nemusí kolemjdoucí zaznamenat veškeré nové produkty. V tom by měl pravdu, nicméně i tak ve vazbě na kolemjdoucího je vyšší pravděpodobnost, že si všimne alespoň části produktů a při opakované cestě kolem uvidí produkty odlišné. Pokud se toto vše spojí i se zvukem, dochází k dalšímu navýšení pravděpodobnosti upoutání pozornosti.

Zmíněné příklady vedou k vysvětlení pojmu digital signage. Jedná se o tabuli, kterou je možno dynamicky měnit. Ideálně bez nutnosti fyzického přístupu k dané tabuli.



Obrázek 1.1: Billboard (zdroj: autor)

1.2 Klientela systému

Využití systému je velmi rozsáhlé. Proto je základna potenciálních provozovatelů velmi široká. Systém lze provozovat např. v:

- malých obchodech
- obchodnich centrech
- na recepcích
- restauracích, jídelnách
- hotelech
- univerzitách
- úřadech

Takto by se dalo pokračovat. Obecně lze ale za vhodná označit všechna místa, která splňují podmínku vysoké návštěvnosti, resp. velkého průchodu lidí. V ideálním případě velkého počtu lidí, pro které je určen přehrávaný multimediální obsah.

1.3 Benefity systému

Systém může nabízet rychlý a pohodlný způsob šíření informací mezi naprostou většinu lidí, která má přístup k jednotlivým zobrazovacím zařízením. Systém:

- dokáže pomoci s navigací lidí - pomocí map či ukazatelů
- může fungovat ve formě vyvolávacího systému - zobrazuje aktuální číslo zákazníka, který se má dostavit na místo odbavení
- lze využít jako marketingový nástroj - součást reklamní kampaně
- lze využít pro zobrazení aktuálních jízdních řádů (za předpokladu, že je přístup k potřebným datům)

Obsah jednotlivých tabulí je možno měnit na základě předem připravených akcí vázaných na kalendář. Nebo lze zobrazovat dynamická (tzv. živá) data. Živými daty je myšleno např. textové zpravodajství, aktuální počasí či aktuální slevy. Další možností je využíti přenosu multimediálního obsahu v realného čase (*streaming*), čímž se otevírají dveře k přenosu videa - např. přenosu sportovního utkání či živého koncertu (*live*). Dosud vyjmenované možnosti předpokládájí pouze jednosměrný tok informací, kdy se předává obsah směrem k pozorovateli. Tok dat může být ale obousměrný a zpětná vazba může reprezentovat:

- dotyky obrazovky - realizace dotazníku / průzkumu - dynamický informační panel
- vstup z klávesnice
- záznam z kamer - obsah se volí na základě vyhodnocení kamer
- ...

1.4 Multimediální obsah

Multimediální obsah, který je systém schopen zpracovat a následně reprodukovat:

- statické obrázky
- dynamické obrázky (animace)
- statické texty

- animované (pohyblivé) texty
- předem připravená videa
- stream videa (živě / na požádání)
- aktivní moduly (počasí, akcie)
- s využitím zpětné vazby - průzkumy, informační systémy, feed[1] ze sociálních sítí

Situace, jenž ve stejný čas je zobrazeno více médií je běžnou praxí. Obrazovka je rozdělena na více částí a každá jednotlivá část má za úkol zobrazovat svůj vlastní obsah - nemusí být vůbec závislý na částech okolních. Přechody mezi jednotlivými obsahy bývají realizovány pomocí animací:

- prolnutí
- přejetí nahoru/dolů/vlevo/vpravo
- další běžně využívané animace
- animace naskriptované dle fantazie tvůrce obsahu

1.5 Šíření obsahu

Obsah se může šířit více způsoby:

- CD / DVD
- USB flash
- FTP - v přehrávači
- FTP - sdílené
- Hierarchický síťový systém

První, dnes již zastaralé, systémy využívaly k distribuci obsahu fyzická přenosná média, jakými jsou CD, DVD či USB flash paměti. Řešení je to sice funkční, pro dnešní dobu ovšem nedostačující. Je to hlavně z důvodu nutnosti fyzicky se dostavit k danému přehrávači a ručně aktualizovat jeho paměť multimediálních souborů.

Výhodnějším systémem distribuce je využití sítě internet spolu s protokolem FTP. Přehrávač má aktivní vlastní FTP server, což přinese bonus ve formě pohodlnější aplikace změn – odpadá nutnost fyzické navštěvy přehrávače kdykoliv při změně obsahu. Nový obsah se na-

[1] Feed je strojově čitelná datová struktura, pomocí níž se přenáší určité informace.

hraje na zmíněný FTP server a přehrávač se synchronizuje automaticky. Jako nevýhoda se dá považovat nutnost aktualizace více FTP serverů přehrávačů mnohdy se stejným obsahem.

Tento problém částečně řeší sdílené FTP, které je na samostatném serveru a přehrávače na něm odposlouchají změny. Pro aktualizaci všech poslouchajících přehrávačů stačí obsah nahrát pouze jednou. Určitým způsobem je to jednodušší, ale všechny přehrávače obsahují totožný obsah.

Komplexnějším systémem, který je schopen reflektovat prakticky všechny požadavky, je hierarchický systém. Díky jeho struktuře jsou veškeré prvky adresovatelné a při nutnosti aktualizovat určitý obsah lze vybrat, jakých všech přehrávačů se má tato změna týkat. Uživatel dodá pouze jednou patřičné zdroje a určí, které přehrávače je mají začít využívat. K samotné distribuci k jednotlivým přehrávačům dojde již automaticky. Obsah se od uživatele na server dostane skrze FTP, častěji pomocí webového rozhraní, popř. s využitím nativního klienta[2]. Nejčasteji se k síťové komunikaci využívají sockety, SOAP či REST API.

1.6 Kategorizace

1.6.1 Rozsah zobrazení

Systém lze dělit dle počtu zobrazovacích zařízení, se kterými spolupracuje. Existují systémy, které spolupracují pouze s jednou obrazovkou. K dispozici jsou ale i systémy, které zvládájí obrazovek více.

Jediné zobrazovací zařízení

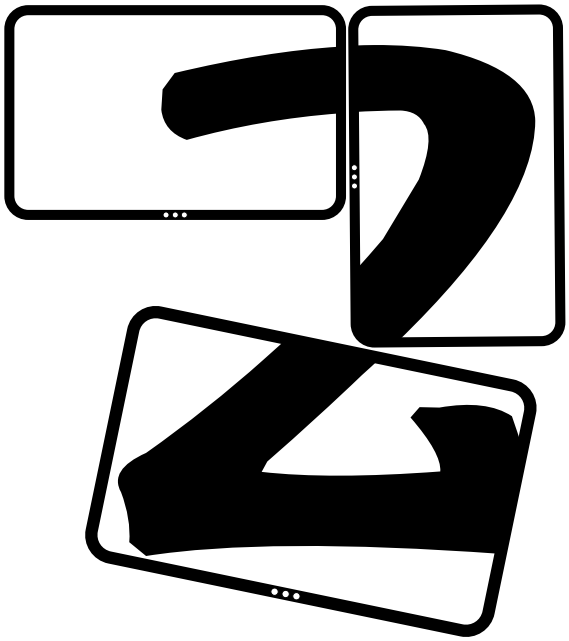
Jedná se o případ znázorněný na obrázku 1.2, kdy se k zobrazení informace použije pouze jednoho zobrazovacího zařízení. Nedochází k dělení obrazu.

Obrázek 1.2: Obsah odpovídá jednomu zobrazovacímu zařízení (zdroj: autor)

[2] Nativní klient je aplikace běžící přímo pod daným operačním systémem. Nepotřebuje pro svůj běh aplikace třetích stran.

Více zobrazovacích zařízení

Jsou situace, kdy si zobrazovaný obsah nevystačí s jedinou obrazovkou. Může jít např. o požadavek na zvýšení obrazové plochy. Může se stát, že aktuální zobrazovací technologie již nedokáže pokrýt potřebu určitých rozměrů zobrazovací plochy při zachování určité jemnosti obrazových bodů. Nastává pak nutnost použití alternativy. Alternativou může být právě více zobrazovacích zařízení použitých v určitém vzájemném uskupení. Vykreslovaný obsah je kreslen přes všechny připojené plochy. Dalším příkladem je umělecký směr, kdy připojené obrazové mohou mezi sebou svírat různé úhly. Nemusí tak jít jen o vodorovné či svislé otočení. Takový případ je znázorněn na obrázku 1.3. Takovéto případy ale mnohdy přinášejí starosti se synchronizací obsahu. Pokud dochází k přehrávání videa přes několik zařízení, musí být kladen důraz na synchronizaci jednotlivých obrazů. Pokud by docházelo při přehrávání k časovým posuvům mezi jednotlivými obrazy, stává se systém prakticky nepoužitelným.



Obrázek 1.3: Obsah přes více zobrazovacích zařízení (zdroj: autor)

1.6.2 Informační zaměření

Systémy se rozlišují dle informačního zaměření. Každé zaměření má své specifické vlastnosti a může vnášet odlišné požadavky. Obecně jsou vžita následující tři odvětví:

- POS (point of sale) - Nákupní místo
- POW (point of wait) - Fronta
- POT (point of transit) - Dopravní/navigační informace

POS (point of sale) - Nákupní místo

Tento bod je považován za místo poskytující informace o nabízeném zboží nebo službách. Může informovat o aktuálně nabízeném zvýhodněném zboží apod. Obecně lze tento bod označit jako výkonnou či realizační složku marketingu, pomocí které je napomáháno k uspokojení potřeb zákazníků.

POW (point of wait) - Fronta

Point of wait, neboli fronta, je část, která řeší šíření informací o čekání na úřadě, lékaře apod. Pro zajištění sofistikovanější obsluhy fronty může být součástí i objednávkový systém. Poté systém může převzít funkci vyvolávání následujících zákazníků, počítat průměrnou dobu čekání a provádět další statistické výpočty.

POT (point of transit) - Dopravní/navigační informace

Point of transit, neboli dopravní/navigační informace, mají za úkol šířit informace o stavu vozovky nebo o dojezdových časech např. na dálnicích. Informovat o dojezdových časech. Tyto systémy mohou být umístěny také na vlakových nádražích i letištích, kde informují o odjezdech vlaků z určitých kolejí, resp. odletech letadel z terminálů.

1.6.3 Faktory ovlivňující obsah

Systémy lze odlišit dle fakturů, na zákadě kterých dochází k šíření a reprodukovaní informací. Faktory ovlivňující informace jsou:

- čas (kalendář)
- kontextové faktory
- relevantnost až ve chvíli zobrazení (reálný čas)

Čas (kalendář)

Předem je určen obsah, který se má přehrávat v daném čase. Popř. je určena posloupnost různých obsahů, která je poté přehrávána v nekonečné smyčce. Posloupnost obsahů je označována jako *mapování na časovou osu* (anglicky *timeline*) a jedná se o hojně využívaný přístup ke skládání obsahu. Blížší specifikace tohoto postupu je popsána v kapitole 1.8.1.

Kontextové faktory

Zobrazovaný obsah není spojen s časem, avšak je neměnného charakteru. To znamená, že obsah je předem znám a jeho relevantnost je neměnná v čase, popř. má nastavenu dobu platnosti. Rozdíl oproti časovému přístupu je v tom, že veškerý obsah je obohacen o kontextové faktory. Tento pojem bude vysvětlen na následujícím příkladu: existuje systém, který má za úkol zobrazit detaily určitého produktu. Systém je rozšířen o schopnost získání zpětné vazby ze svého okolí – v našem případě ve formě elektronické čtečky např. RFID kódů. Díky tomuto je systém informován, že někdo má zájem o zobrazení produktu opatřeného kódem. Systém požádá jiný kooperující systém, aby mu vrátil patřičné informace pro přečtený kód, a tyto informace poté v předem určeném tvaru zobrazí.

Z uvedeného příkladu je zřejmé, že je kladen důraz na databázi znalostí. Databáze musí být optimalizovaná pro vyhledávání dle patřičných kontextových faktorů – indicií.

Dalším obdobným vstupem může být obsažená klávesnice, dotyková obrazovka či kamera s podporou snímání gest. Systém může také reagovat na zvuk - podpora hlasového ovládání je v dnešní době velmi skloňované téma. Dalším způsobem je zahrnutí různých čidel. Od nejjednodušších světelných, až po čtečky RFID, QR kódů, NFC aj. V tomto případě se při přečtení určitého kodu může zobrazit konkrétní produkt s bližšími informacemi o něm.

Obecně je systém napojen na jiný systém, který dokáže autorizovat uživatele a na základě toho zobrazit relevantní obsah. Jednotlivý obsah (informace, reklama) je označena klíčovými slovy a každý uživatel (entita[3]) má své preferované – při autorizaci pak dochází k vyhledávání nejpřesnějších výsledků.

Určité realizace mohou mít i formu informačních terminálů. Může se jednat i o bankomaty apod. Možnosti vzniku tohoto obsahu je možno nalézt v kapitole 1.8.2.

Relevantnost až ve chvíli zobrazení (reálný čas)

Informace reprodukované systémem nebývají předem známy – nelze je zařadit do konrétního scénáře a vytvořit statické video. Jedná se zpravidla i o informace, které mívají krátkou dobu platnosti. Typickým příkladem jsou aktuální ceny akcií či komodit, aktuální počasí s předpovědí nebo dokonce i konzumace novinek ze sociálních sítí apod. Systém musí podporovat způsob komunikace s dalším sub-systémem, který má schopnost zjištění požadovaných informací. Více o způsobu tvorby tohoto obsahu v sekci 1.8.3.

[3] Entitou je myšlena datová reprezentace konzumenta obsahu.

Kombinace

Existují systémy, které spadají ryze do jedné kategorie, ale jsou i systémy, které dokáží pokrýt více (i všechny) kategorie. V posledním případě se jedná o velice komplexní systémy.

1.6.4 Dle vlastníka

Velkou roli při výběru systému hraje fakt, kdo je vlastníkem systému po získání licence daného systému. V tomto případě nejde o určení konkrétní fyzické či právnické osoby. Jde o to, zda systém bude zcela ve vlastnictví inzerenta či se jedná o systém sdílený.

Vlastní inzerent

Výhodou tohoto systému pro inzerenta je, že celý systém je jeho a má výhradní právo na jeho správu i užití. Po celou dobu fungování systému určuje jeho majitel, co se bude přehrávat. Nevýhodou je, že se sám majitel musí postarat o distribuci jednotlivých koncových přehrávacích zařízení na správná místa - z toho vyplývají další náklady jako pronájmy prostor, energie a servis zařízení. Čím více má tento systém tendenci se rozšiřovat, tím se stává nákladnějším a náročnějším na údržbu.

Sdílený

Systém vlastní jeden subjekt, který se stará o celou infrastukturu a jednotlivé plochy nabízí k pronájmu inzerentům. Inzerent v tomto případě nemusí řešit žádné problémy s infrastrukturou. Pouze se dohodne s pronajímatelem, kdy a na jakých místech chce, aby docházelo k zobrazení jeho obsahu. Následně pak za sjednanou částku dojde k zobrazení obsahu.

V tomto přístupu se jednotlivé obsahy inzerentů dělí o časové pásmo, resp. soupeří dle určitého kontextu, kdo splní přesněji požadavky diváka.

1.6.5 Dle ovládání

Systém může podporovat i určitý systém ovládání. Nemusí jít pouze o jednosměrný tok dat. Existují tak systémy:

- bez zpětné vazby
- se zpětnou vazbou

Bez zpětné vazby

Jedná se o zařizení s jednostranným tokem dat. Umožňuje informace reprodukovat divákům. Neexistuje však cesta, kterou lze získat zpětnou vazbu od diváků systému.

Se zpětnou vazbou

Systém obsahuje kromě zobrazovacího zařízení i zařízení vstupní. Vstupní zařízení generuje události, na které lze reagovat předem připraveným způsobem. Např. pohybové čidlo zazmená pohyb a tím se aktivuje přehrávání obsahu. Motivací pro vypnutí přehrávání při nepřítomnosti diváka, snížení nákladů na spotřebu energie.

Příklady způsobu zpětné vazby:

- dotyky obrazovky - realizace dotazníku / průzkumu - dynamický informační panel
- vstup z klávesnice
- kamery
- mikrofon
- GSM (průzkum: zavolejte na čislo x, pokud souhlasíte s..) / SMS

1.7 Fyzické prvky systému

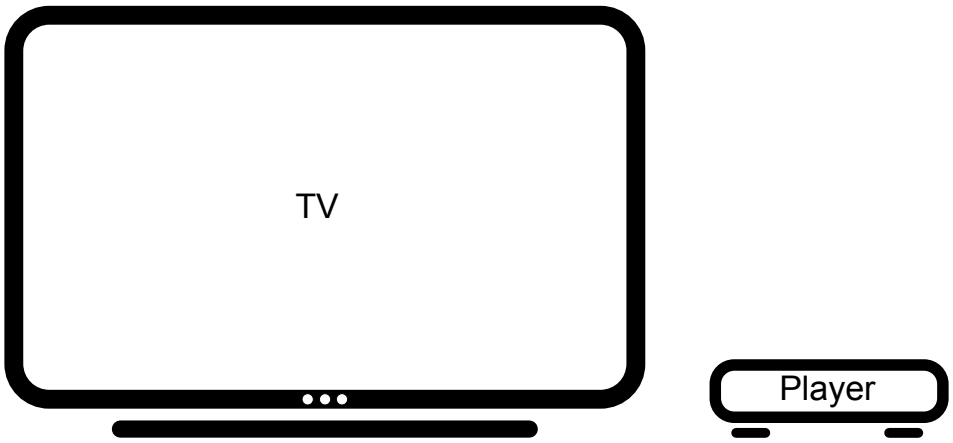
I když se to implementace od implementace liší, jsou hardwarové prvky, které jsou totožné napříč různými řešeními.

1.7.1 Konečný uzel

Nejdůležitějším prvkem v systému je zobrazovací zařízení, kde se bude přehrávat požadovaný obsah. Toto je nejpodstatnější část systému, protože se jedná o základní vlastnost, kvůli

které se systém vůbec navrhuje. Tento prvek je reprezentován např. panelem LCD, plasmovou obrazovkou, projektorem apod. K obrazovce musí být zpravidla připojen přehrávač (obr. 1.4). Přehrávač může mít podobu stolního počítače, miniaturního počítače jako např. *Raspberry PI*[4] či zcela vlastního řešení. Tyto dva prvky mohou být sloučeny v jeden. Pak reprezentantem této realizace může být tablet. Nezřídka kdy je použito i vlastní řešení, ale vždy se jedná o kombinace počítače se zobrazovacím zařízením. Zařízení potřebuje paměťové médium k ukládání zobrazovaného obsahu. U zařízení je nutné zajistit přívod elektřiny a rychlého připojení k internetu. Potřebná rychlost internetového připojení se odvíjí od požadavků na náročnost přehrávaného obsahu.

Hlavní funkcí přehrávače je renderování obsahu. Může být ale obohacen i o funkce pro správu obsahu i zařízení samotného. I když některé přehrávače dokáží fungovat zcela autonomně, tzn. sami dokáží udělat rozhodnutí o tom, co se bude přehrávat, je třeba zajistit určitou synchronizaci s ostatními. Síť přehrávačů může obsahovat několik desítek či stovek zařízení, a tak musí být zajištěna jejich správná komunikace. Samozřejmostí je patřičný stupeň zabezpečení, aby obsah zobrazovaného materiálu mohla ovlivnit pouze kompetentní osoba.



Obrázek 1.4: TV + přehrávač (zdroj: autor)

1.7.2 Administrace

Nad samotnými přehrávači je multimediální/distribuční server. Ten je určen k uchovávání multimediálního obsahu, poskytování obsahu na požádání, zpřístupnění možnosti konfigurace multimediálního obsahu apod. Server musí obsahovat i určitý systém zabezpečení, aby nevydal obsah každému, kdo vznese požadavek. Naopak musí zajistit, aby byl přístup k obsahu, resp. modifikačním funkcím, komukoliv cizímu odepřen. Měl by obsahovat seznam uživatelů, kteří mohou přistoupit a mohou systém sledovat, resp. konfigurovat. Server

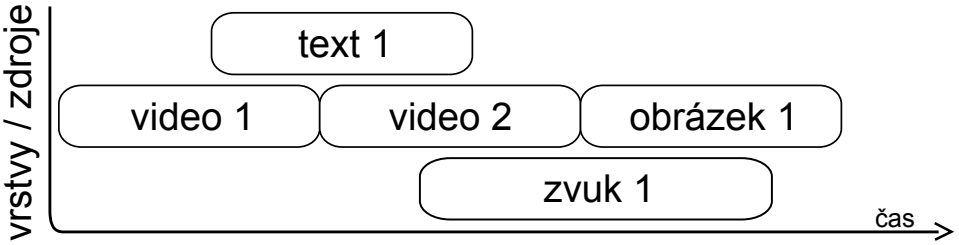
[4] Raspberry PI je malý jednodeskový počítač o velikosti platební karty.

není nic jiného, než opět počítač s dostatečně velkým paměťovým médiem a je opatřený dostatečně propustným připojením k internetu. Zpravidla jsou servery provozovány ve speciálních serverovnách, kde jsou zajištěny optimální podmínky pro jejich běh.

1.8 Tvorba/správa obsahu

1.8.1 Timeline - časová osa

Jedná se o jeden ze způsobů, jakým lze vytvářit obsah pro přehrávače. K dispozici je renderovaná oblast (oblast později zobrazená divákům), která má být vyplněna obsahem. Obsahem, jak bylo zmíněno výše, mohou být obrázky, videa, text atd. Veškerý tento obsah je popsán pomocí parametrů. Parametry je vyjádřena například pozice umístění obrázku v renderované oblasti, velikost, průhlednost atd. Tyto parametry se poté mapují na časovou osu. Při spuštění přehrávání se aktuální stav posouvá po časové ose a aplikují se aktuální změny parametrů, které jsou na ni uvedeny. Tento způsob je založen na čase, kdy se určí jedna pomyslná časová osa, na kterou se váží určité události - změny parametrů.



Obrázek 1.5: Ukázka časové osy (zdroj: autor)

Obrázek 1.5 znázorňuje, jak taková časová osa může vypadat. Uživatelské ovládání vkládání obsahu a přesouvání v čase bývá zpravidla řešeno pomocí myši. Pomocí myši lze obsah uchopit a přesouvat ho po časové ose. V některých případech lze přesouvat i vertikálně – mezi vrstvami. Toto slouží k nastavení hloubky, pomocí které se při vykreslování docílí správného vrstvení obsahu – korektního překrývání.

1.8.2 Kontextový obsah

Zde jsou definovány dva prostory. Prvním prostorem jsou kontextové faktory, pomocí nichž je snaha vyhledávat. Druhým prostorem jsou pak samotné informace. Hlavním předmětem mapování kontextu je lineární zobrazení z prostoru kontextových faktorů (neboli indícií) na prostor samotných informací. Pokud zobrazení existuje, vložené indicie byly správné a databáze znalostí zná odpověď. V opačném případě neexistuje známá odpověď – špatný vstup / neexistence reakce na podnět.

Vytvoření tohoto systému je založeno na základě určení indícií, na které je snaha reagovat, např. kódů produktů, hlasových příkazů, předpisů pro obrazové rozpoznání člověka/obličeje atd. Informace pro určené indície mohou být např. detaily produktů s produktovými fotografiemi, reakce na hlasové příkazy, informace určené pro muže/ženy atd.

1.8.3 V reálném čase

Aby fungoval systém v reálném čase, musí být napojen na externí systém, který dokáže zajistit potřebné informace. Pro příklad, zmíněný externí systém dokáže poskytnout informace z finanční burzy, což zajistí přístup k aktuální ceně akcií a komodit. Externí systém může být také zastoupen fyzickou osobou, která píše zprávy o aktuálním dění sportovního přenosu. V neposlední řadě může komunikovat i se systémem, skrze který lidé dobrovolně sdílejí své fotografie. Díky tomu je náš systém zásoben sérií fotografií napříč celým světem.

2 Rešerže existujících systémů

2.1 Xibo - Digital Signage

Označení Xibo [6] nese systém, jehož autorem jsou Daniel Garner, James Packer a Alex Harrington. Počátek systému je v roce 2004, kdy vznikal jako ročníková práce v posledním roku studia na vysoké škole. Od té doby prošel řadou změn a nyní je částečně vydáván pod licencí AGPLv3. Díky této licenci autor musí zpřístupnit veškeré zdrojové kódy vydané pod touto licencí všem uživatelům, kteří o to projeví zájem. Zdrojové kódy jsou tedy k dispozici komukoliv ve veřejném repozitáři na githubu [7]. Díky tomuto kroku může kdokoliv nahlížet, jak byla řešena daná problematika. Znalá osoba může vyhledávat chyby v systému a navrhovat řešení, případně chybu i odstranit. Záplatu chyby pak může nahrát zpět do repozitáře. Pokud bude záplata přijata, bude zařazena do nové verze systému.

Návrh je autory rozdělen na dvě komponenty: samotný přehrávač a CMS (Systém pro správu obsahu).

- Přehrávač
- CMS (Systém pro správu obsahu)

  Vlastní instance

  Cloud

2.1.1 Přehrávač

V době psaní této diplomové práce jsou k dispozici tři nativní přehrávače:

- Microsoft Windows
- Android
- Ubuntu (vývoj zastaven)

Přehrávač pro Microsoft Windows je napsán v technologii .NET a je dodáván zcela zdarma. Přehrávač pro Android je k dispozici za poplatek za každé zařízení či rámcově dle smlouvy.

Jeho zdrojové kódy jsou neveřejné. Posledním je linuxový přehrávač určený pro systém Ubuntu. Jeho vývoj byl však pozastaven.

V tabulce 2.1 je porovnání prvních dvou výše zmíněných přehrávačů.

2.1.2 CMS - Systém pro správu obsahu

Pomocí této komponenty je zpřístupněno veškeré ovládání a správa přehrávačů.

- Knihovna médií (obrázky, videa)
- Knihovna dynamických médií (RSS čtečka, hodiny apod.)
- Kalendář – plánování přehrávání obsahu ve vazbě na kalendář
- Správa přehrávačů a jejich statistika
- Správa uživatelských rolí

Systém pro správu obsahu je napsán v technologii PHP. Pro ukládání dat využívá databázi MySQL. Z toho vyplývá, že jde o webovou aplikaci. Přístup k tomuto redakčnímu systému lze uskutečnit dvojí cestou.

Vlastní instance

Celé CMS lze volně - bezplatně stáhnout a nainstalovat na vlastní počítač či server. Jediným požadavkem na cílový počítač je nutnost podpory webového serveru PHP a databáze MySQL. Vzhledem k tomu, že se jedná o jedno z nejrozšířenějších řešení pro webové aplikace, tak podpora ze strany těchto dvou technologií je značná. Technologie spolupracují s počítači s operačním systémem Windows, Linux a Mac.

Vlastní instalací se docílí toho, že správa celého CMS bude jen v režii toho, kdo provozuje, resp. spravuje, tento daný počítač. Autor systému Xibo nemůže nijak zasáhnout do provozu systému – např. nemá přístup k datům provozovatele vlastní kopie. Naopak nevýhodou je nutnost zabezpečit CMS "svépomocí", např. postarat se o patřičné zálohy pro případ výpadku apod.

Cloud

Druhou možností je využít centralně řízené verze systému Xibo – neboli cloud. Ten je ovšem součástí placené verze systému a pro jeho provozování je potřeba zakoupit patřičnou licenci. Z pohledu uživatelského ovládání je to totožné, jako v případě vlastní instance. Od-

Tabulka 2.1: Porovnání Xibo přehrávačů pro Windows a Android

Funkce	Xibo pro Windows	Xibo pro Android
Rozvržení obrazovky	Ano	Ano
Prioritní plánování	Ano	Ano
Video	Ano	Ano
Flash	Ano	Ne
Obrázky	Ano	Ano
PowerPoint	Ano	Ne
Text	Ano	Ano
RSS	Ano	Ano
Webová stránka	Ano	Ano
Vnořené HTML	Ano	Ano
DataSety	Ano	Ano
Nastavitelný obrázek pozadí	Ano (pouze formát jpg)	Ano
Statistiky médií	Ano	Ano
Statistiky rozvržení	Ano	Ano
Inventář reportů	Ano	Ano
Obnova souborů	Ano	Ano
Interaktivní obsah (v přípravě)	Ne	Ne
Socket Listener	Ne	Ne
Client Runtime Information Screen	Ano	Ano
Změna médií offline pomocí USB klíčenky	Ne	Ne
Full Compositing (overlapping regions)	Ne	Ano
Průhlednost webových stránek	Ne	Ano
Průhlednost videa	Ne	Ano
Průhlednost obrázků	Ne	Ano

padá ale starost o fyzický počítač. Veškerá údržba, aktualizace i zabezpečení je nakoupena formou služby u autorů systému Xibo.

2.2 Signagelive

O tento systém [12] se stará společnost, která působí ve více než 40ti zemích. Hlavní sídla má tři – ve Velké Británii, Spojených státech amerických a v Singapuru. Poskytují velmi rozsáhlá řešení pro různé potřeby zákazníků. Specializují se na široké spektrum od maloobchodů, přes rychlé občerstvení, až po šíření informací ve firemním komplexu budov.

Součástí systému je několik druhů přehrávačů s různými úhlopříčkami obrazovek. Zobrazovaný obraz lze rozdělit na více obrazovek, pomocí čehož se zvýší zobrazovací plocha. U jednotlivých ploch je možnost nezávisle nastavit, co a kdy se má zobrazit ve vazbě na kalendář. Obrazovky tvoří tzv. layouty[1] – jedna obrazovka může být rozdělena na více layoutů. Tím se docílí toho, že jedna obrazovka je schopna zobrazit více různých modulů najednou. Modulem se nazývá jedna zapouzdřená funkčnost, která má jednu cílovou akci, např. zobrazení hodin, přehrávání videa či zobrazení produktu.

2.2.1 Maloobchody

Řešení pro maloobchody umožňuje definovat obchodní plán a tím upřednostňovat konkrétní produkty. Díky interaktivitě, neboli zpětné vazbě, může zákazník ovlivnit zobrazené produkty. Sám může listovat v katalogu a vybrat, o jaký detail produktu má zájem. Díky tomu se šíří bližší informace v oblasti zájmu zákazníka, čímž se podpoří pravděpodobnost úspěšného prodeje.

2.2.2 Rychlé občerstvení

Zajímavou oblastí je rychlé občerstvení. Pro tuto oblast poskytují nejen informační tabule s nabízenými produkty, ale také je obohacena i o mobilní aplikaci. V mobilní aplikaci, kterou si zákazník může spustit na svém mobilním zařízení, jsou k dispozici další informace o produktech. Typickými informacemi v oblasti gastronomie je složení jednotlivých produktů, jejich nutriční hodnoty nebo alergeny. Díky tomuto se v jednom řešení zapouzdří reklamní banery za výlohou, informační banery s jídelním lístkem u výdejního místa až po dílčí informace o jednotlivých produktech na chytrých zařízeních zákazníků.

[1] Layout je označení pro předem určené rozložení prvků na obrazovce.

2.2.3 Firemní interní komunikace

Komplexním firemním řešením je v nabídce rozsáhlý systém pro celou budovu. Součástí je navigační panel s mapou budovy a běžnými prvky jako informačními panely. Zajímavým prvkem je virtuální recepce, která umožnuje evidovat příchozí návštěvy. Při zaznamenání nového návštěvníka dokáže informovat na informačních panelech či e-mailem. Informační panely lze využít i pro reprezentaci statistických dat či výsledek analýzy tzv. "big data". Systém umí přenášet i živý obraz, takže televizní vysílání či konferenční hovor není neřešitelnou problematikou. V neposlední řadě obsahuje i funkčnost pro rezervaci jednotlivých sdílených místností, typicky konferenčních, nebo zasedacích místností.

2.3 Screenly

Screenly [11] je název systému z rodiny digital signage vyvinutý americkou společností Wire-Load Inc. Řešení staví na přehrávačích realizovaných pomocí Raspberry PI v kombinaci s PC serverem. Nabízí jak variantu open-source, tak komerční řešení. Jejich rozdílné vlastnosti jsou zachyceny v tabulce 2.2. Komerční řešení jsou pak nabízeny pod různými licencemi dle rozsáhlosti systému.

Tabulka 2.2: Porovnání podporovaných funkcí mezi licencemi Screenly

Funkce	Screenly OSE	Screenly Pro
Příprava pro Raspberry PI	Ano	Ano
Full HD	Ano	Ano
Obrázky, video a web	Ano	Ano
Webový správce	Ano	Ne
Funkce jednoduché instalace obrazovky	Ne	Ano
Správa více přehrávačů z jednoho místa	Ne	Ano
Podpora vzdálené správy	Ne	Ano
Cloudové úložiště	Ne	Ano
Automatické video dekódování	Ne	Ano
Dashboard	Ne	Ano
Upozornění	Ne	Ano
Cachování obsahu	Omezené	Ano
Automatické aktualizace	Ne	Ano
Podpora systémových vylepšení	Ne	Ano
Podpora	komunita	komerční

2.3.1 Dashboard

Jelikož toto řešení podporuje zobrazení webové stránky, je vhodné pro zobrazení statistik, grafů, kalendáře akcí apod. produkované řešeními třetích stran. Nabízí dobrou správu zobrazovaného obsahu.

2.3.2 Školství

Screenly cílí i na školství. Díky relativně dostupné cenové hladině může školská instituce vybavit své pracoviště obrazovkami, pomocí kterých může šířit informace mezi své zaměstnance a studenty. Tento systém mohou využít i sami studenti k prohloubení znalostí o způsobu práce s podobnou problematikou, a to díky uvolněnému zjednodušenému open-source řešení.

2.3.3 Restaurace

Systém má implementovaný zajímavý pohled pro práci se scénářem, ve kterém lze nastavit, že v určitý čas má vstoupit konkrétní obsah. Typicky aktuální polední menu má smysl ukazovat pouze přes poledne. Systém dokáže zpracovat informaci, která rozděluje obsah nejen na obědová menu, ale i snídaňová menu či dokonce víkendová. Při správném nastavení lze docílit vcelku pěkného autonomního přístupu systému, který se sám automaticky přizpůsobuje denní době.

2.3.4 Maloobchodní síť

V maloobchodní síti lze systém využít pro tvorbu prodejních kampaní. Vzájemně se mohou prolínat videa s obrázky v rámci rozlišení obrazovky full HD. Kampaně lze oživit i o vstupy s živými daty ve formě webových stránek. Tím lze docílit zobrazení aktuálního počasí, zpráv a dalších. Implementovaný kalendář akcí dovoluje nejen nastavit počátek a konec pro danou kampaň, ale i omezit zobrazení kampaně např. na každý pátek po patnácté hodině.

2.4 Embed Signage

Rozsáhlé řešení skrývající se pod názvem Embed Signage [10] je z pohledu nabízených funkcí velmi obsáhlé. Systém může ovládat prakticky neomezené množství lidí, přičemž každému lze vyčlenit přístupnou část. Systém má také propracovanou správu přehrávačů, kdy je přehledně vidět aktuální stav přehrávačů a lze je pohodlně ovládat. Umístění jednot-

Tabulka 2.3: Podpora platforem systémem Embed Signage

Platforma	Podpora
iOS (iPhone, iPad, iTouch – iOS6+)	Ano
Android (Tablet, Smartphone and Mini PC – Jelly Bean 4.2+)	Ano
Windows (Windows 7, Windows 8 and Windows 8.1 (32bit))	Ano
Mac (10.7+ (32bit))	Ano
ONELAN Digital Signage Media Players (V9.3.7+)	Ano
Linux	Ne

livých zařízení lze zobrazit na mapě. Zobrazovaný obsah je tvořen responzivně – dokáže se automaticky přizpůsobit velikosti displaye, na kterém je zobrazen.

2.4.1 Interaktivita

Systém nabízí práci se zpětnou vazbou. Fungují prvky klasických dotyků obrazovky, stejně tak lze využít i předpřipravených gest – namátkou *swipe up*[2], *swipe left* atd. Pomocí těchto činností lze vytvořit prezentaci, kdy je možno využít dialogových oken[3] pro zobrazení doplňujících informací či definovat více oken. Divák si mezi nimi může sám, dle svého uvážení, přepínat.

2.4.2 Cloudové řešení

Jedná se o cloudové řešení, a tak jeho správa je dostupná ze všech částí světa (za předpokladu, že je k dipozici internetové připojení). Veškerý obsah je šířen skrze toto úložiště.

2.4.3 Podporovaná zařízení

Velkým plusem tohoto systému je velká podpora nativních mobilních platforem a části desktopových. V tabulce 2.3 jsou zaneseny podporované platformy.

[2] Swipe je označení pro gesto provedené prstem na dotykové obrazovce.

[3] Dialogové okno je označení pro okenní aplikaci oznamující určitou konkrétní věc. Většinou se očekává návratová hodna souhlasu, nebo nesouhlasu.

2.4.4 Plánování

Plánování zobrazovaného obsahu se váže na čas a konkrétní přehrávač. Je umožněno nastavit správný scénář ve správný čas na správné místo. Dlouhodobé scénáře lze přepsat krátkodobým scénářem s vyšší prioritou. Pod tím si lze představit dlouhodobou kampaň přepsanou zítřejší speciální nabídkou. Až zítřejší nabídky skončí, vrátí se automaticky opět dlouhodobá kampaň.

2.4.5 Pluginy a widgety

Pokud se stane, že systém neobsahuje určitou funkci, lze jej o ni rozšířit. Embed Signage podporuje rozšíření formou pluginů a widgetů. Mezi implementované patří např. podpora RSS čtečky, IPTV, sociální sítě Twitter, zobrazení tabulek z Microsoft Excel, rezervaci místností aj. Systém rozšířit o vlastní pluginy bohužel nejde, nicméně v tomto případě nabízí kontaktování jejich podpory a pořebnou funkcionalitu jsou schopni sami implementovat. O co systém naopak rozšířit lze jsou vlastní typy písem, nebo-li fonty.

2.5 Wirespring

Americká společnost *WireSpring Technologies, Inc.* [9] nabízí více vzájemně propojených řešení. Kromě klasického digital signage zajišťují LED billboardy, interaktivní kiosky a tzv. chytré zařízení postavené na *Machine-to-Machine* (zkráceně M2M) a *Internet Of Things* (zkráceně IoT).

2.5.1 Digital signage

Podobně jako ostatní nabízí toto přešení pro širokou škálu zákazníků – od korporátních společností, hotelů a bank, přes obchody, až po školství a nemocnice. Celý systém v sobě kombinuje jednotlivé moduly. Jedním z modulů je centrální redakční systém, pomocí něhož se spravují jednotlivé připojené zařízení a jejich obsahu. Systém dovoluje členit jednotlivá zařízení do skupin a ty poté ovládat zvlášť. Šířený obsah koresponduje s časem, díky čemuž centrální redakční systém dovoluje řídit, jaký obsah se má kde zobrazit – tím se řeší šíření správného obsahu ve správnou dobu na správné místo.

2.5.2 LED billboardy

Místo klasických televizí či počítačových monitorů jsou v nabídce LED billboardy. Jejich nespornou výhodou je, že jsou vyráběny o rozměrech v řádu metrů až desítek metrů. Pokud

se takovéto rozměry zkombinují s velkou svítivostí jednotlivých diod, vzniká dobrý kandidát pro velkovní využití. Zařízení lze kombinovat s digital signage systémem, takže lze řídit zobrazovaný obsah.

2.5.3 Interaktivní kiosky

Jedná se o samostatné zařízení pracující v režimu *na požádaní*. Zařízení má k dispozici funkcionalitu, která zajistí zpětnou vazbu od diváka a tím se zobrazovaný obsah ovlivňuje. Kiosek má v sobě implementován řadu obrazovek a formulářů spolu s popisem možných uživatelských akcí. Na tomto základě je pak kiosek schopen nabídnout odezvu na divákovy akce.

2.5.4 Chytré zařízení

Další z nabízených řešení definuje komunikační rozhraní pro podporu komunikace stroje se strojem. Navržené rozhraní je nutno zahrnout již při návrhu nově vyvíjeného hardwarového zařízení. S komunikačním rozhraním lze zařízení zařadit do rozsáhlejšího systému nabízejícího určité možnosti, jakými jsou:

- vzdálené sledování chodu
- vzdálené ovládání
- diagnostika zařízení
- servisní podpora
- správa softwarového vybavení
- správa obsahu
- unifikovaný přístup k centrálnímu datovému skladu
- možnost tvorby vlastního dashboardu
- datové analýzy a reporty

3 Specifikace vytvářeného systému

Na základě předchozí analýzy již existujících systémů byl proveden návrh vlastního řešení. Systém navržený v této práci se snaží pokrýt zájem malých podniků, které by podobný systém uvítaly. Avšak trhem nabízená řešení jsou až moc komplexní či ceny za licence se pohybují mimo jejich finanční možnosti. Systém je navrhován s přihlédnutím na fakt, aby jeho ovládání bylo maximálně intuitivní a práci s ním zvládla každá předem seznámená osoba.

Typickým představitelem provozovatele systému je malý, regionální obchod s jednou provozovnou, do kterého se umístí dvě až tři zobrazovací zařízení, na kterých se promítne např. provozní doba obchodu spolu s reklamou na právě prodávané zboží. Dalšími příklady pak jsou regionální informační městská centra či malý živnostníci s vlastní pobočkou pro setkávání s jejich klienty.

3.1 Struktura aplikace

Aplikaci je třeba členit na části dle jejich jednotlivých zaměření. Možné členění je navrženo v této kapitole a odpovídá jednotlivým podnadpisům.

3.1.1 Přehrávač

Cílem aplikace má být zajištění plynulého zobrazení předem definovaného obsahu. Definice obsahu má být formou scénáře. Přehrávač má být schopen konzumovat celý scénář a generování výsledného výstupu provádět v reálném čase. Přehrávač je také schopen přijmout jedno video, které bude přehrávat stále dokola. O všem, co přehrávač provádí, musí informovat synchronizační-datový server. Informovat musí ve chvíli, kdy se daná akce stala. Přehrávač je schopen konzumovat příkazy od synchronizačního-datového serveru. Příkazy se týkají konfigurace přehrávače a ovládání přehrávání.

Přehrávání lze ovládat příkazy *play*, *pause*, *stop* a *reset*.

Play spustí přehrávání obsahu. Pause pozastaví přehrávání obsahu. Stop pozastaví přehrávání obsahu a vrátí přehrávací hlavu na počátek obsahu. Reset vrátí přehrávací hlavu na počátek a v případě, že před zavoláním příkazu byl obsah přehráván, bude se přehrávat dále, ale již od začátku.

Konfigurace přehrávače má být určena k nastavení parametrů:

- Název přehrávače
- Úhel natočení zobrazovací obrazovky přehrávače
- Obsahu k přehrávání

Výše zmíněná informace o konfiguraci úhlu natočení zobrazovací obrazovky přehrávače znamená, že přehrávač je schopen pootočit celý vykreslovaný obsah o libovolný úhel. Toto je třeba pro konfiguraci módu zobrazování obsahu přehrávaře v tzv. "landscale"(na šířku), popř. "portrait"(na výšku). Pro mód na šířku se volí nulový úhel posunu, pro mód na výšku pak pootočení o 90 stupňů. Toto nejsou jediné možnosti, které tímto lze docílit. Je možné zobrazovací zařízení pootočit např. o 45 stupňů a pomocí tohoto pootočení zajistit, že obsah bude vykreslován stále rovnoběžně s podlahou tak, jak je pro člověka obvyklé.

Vykreslování scénáře je konfigurováno pomocí bezrozměrných normovaných jednotek (více o jednotkách v kapitole 3.1.4). Aktuální rozměry vykreslované plochy se převedou na bezrozměrnou normovanou jednotku. Vykreslovaný obsah se snaží maximálně zaplnit vnitro plochy tak, aby zachoval poměry stran scénáře a nepřetékal mimo. Přináší to výhody ve formě zachování kompatibility mezi např. 16:9 scénáři s 4:3 obsazovkou. Vykreslující jádro automaticky obohatí vykreslovanou oblast o jednobarevné pruhy v horní a dolní (resp. v levé a pravé) části. Tím zajistí vykreslení bez přetékání.

3.1.2 Synchronizační-datový server

Server zastává synchronizační funkci jednotlivých částí systému – tzn. veškerá komunikace má být směřovaná skrze tento uzel. Server zajistí ukládání veškerých důležitých dat. Jako důležitá data jsou brány:

- Informace o přehrávačích
- Historie provedených akcí v přehrávači
- Uložené scénáře
- Zdrojové soubory

Server také udržuje seznam uživatelů. Server také dokáže ověřit nově příchozí dotaz, zda pochází od některého z definovaných uživatelů. Pokud tomu tak není, dotaz odmítne. Server také zpřístupňuje funkcionalitu pro přiřazení rolí uživatelům. Tímto lze omezit pravomoce.

Vytvoření video souboru na základě scénáře

Server má také zvládat vytvořit jedno samostatné video na základě již vytvořeného scénáře. Toto je pro případ, kdy z nějakého důvodu nelze nasadit aplikaci *přehrávač*. Tuto skutečnost pak lze obejít pomocí tzv. streamu vytvořeného videa dle scénáře. Stream videa v dnešní době umí převážná většina předních programů pro přehrávání video souborů, které bývají zahrnuty již v základních distribucích operačních systémů.

Správa souborů

Server má umožnit správu souborů – zdrojů pro jednotlivé scénáře. Jednotlivé soubory mohou být řazeny do složek, které jsou uživatelsky definovatelné. Musí být umožněn upload souborů na server spolu s možností smazání, přejmenování a přesunutí souboru v rámci serverové struktury.

Veškeré přípustné soubory pak budou přístupné z editoru scénáře.

3.1.3 Správce přehrávačů

Správce přehrávačů musí umět zobrazit seznam všech přehrávačů. O jednotlivých přehrávačích bude informovat, zda jsou online a jaký obsah přehrávají. Musí zpřístupnit funkce pro ovládání a nastavení přehrávačů. Správce také dokáže získávat informace, co aktuálně zpracovává přehrávač formou *logů*. Skrze poskytované rozhraní lze nastavit i nový scénář.

Správce přehrávačů má k dispozici seznam scénářů. V případě, že přihlášený uživatel má dostatečná práva, smí scénáře měnit pomocí editoru scénáře (více informací o editoru scénáře v kapitole 3.1.4).

3.1.4 Editor scénáře

Musí umět konzumovat již vytvořený scénář a zreplikovat ho jako výchozí konfiguraci editoru.

Layout

Editor má obsahovat následující prvky:

- horní lištu se základními odvládacími prvky
- náhled editovaného scénáře – možnost přehrát scénář

- seznam zdrojových souborů s možností filtrování
- časovou osu pro umístění jednotlivých zdrojů ve vrstvách
- menu pro editaci jednotlivých prvků scénáře

Rozvržení editoru je následovné. Horní lišta obsahuje tlačítka pro základní ovládání editoru. Těmi jsou uložení a zrušení editace bez uložení. Obě tlačítka vedou na seznam uložených playlistů. Dále jsou obsažena tlačítka pro spuštění a zastavení přehrávání náhledu.

Tlačítky z horní lišty ovládaný náhled scénáře je oblast, do které bude vykreslován přehrávaný scénář.

Seznam zdrojových souborů obsahuje všechny soubory, které obsahuje server. Pro snažší orientaci bude zahrnut textový filtr, který dokáže filtrovat i pomocí časti názvu souboru. Soubory budou řazeny v seznamu, kde každý řádek reprezentuje jeden soubor. Každý řádek obsahuje název souboru a obrazový náhled (je-li k dispozici). Dvojtým poklepáním myší na řádek se daný zdrojový soubor přidá do poslední vrstvy na časové ose.

Časová osa je pole o dvou dimenzích, kde první dimenzí jsou vrstvy a druhou dimenzí je čas. Obě vrstvy obsahují nenulový průnik, který určuje, v jakém čase začne aktivně figurovat daný zdroj v přehrávaném scénáři a kdy se ukončí. Na vrstvu je nahlíženo jako na jeden zdrojový soubor. Vrstvy jsou vykreslovány v pořadí, v jakém jsou umístěny na časové ose shora dolů. Může se stát, že se na obrazovce zcela překryje jeden soubor jiným. Pro každou vrstvu lze nastavit pozici levého horního rohu a jeho šířku, resp. výšku. Veškerá tyto poziční nastavení probíhají v bezrozměrných jednotkách v intervalu <0; 1>. Více o bezrozměrných jednotkách v další části této kapitoly. Pořadí vrstev je uživatelsky editovatelné. Pomocí systému *drag and drop* lze vrstvu chytit levým tlačítkem myši a přetáhnout na místo jiné vrstvy. Tím dojde k odsunu nižších vrstev ještě níže.

Průnik dvou zmíněných dimenzí, tedy aktivní zobrazení zdroje, lze také pomocí levého tlačítka myši a systému *drag and drop* posouvat a tím měnit čas, kdy má dojít k aktivaci a deaktivaci zdroje. Posun je ohraničen nulovým časem zleva a celkovou délkou scénáře zprava. Zdroj obsahuje i možnost nastavit délku jeho přehrávání.

Bezrozměrná normovaná jednotka <0; 1>

Tato jednotka je určena pro práci s pozicováním na obrazovce, přičemž obsah obrazovky není mapován vůči rozlišení, ale vůči poměru. Tento postup byl zvolen z důvodu odproštění se od konkrétních rozměrů. Díky tomuto pak samotný scénář nemusí být seznámen s rozlišením konečného zařízení, pro které je určen. Rozhodnutí o správné volbě zdrojových souborů je na uživateli. Při tvorbě scénáře musí uživatel přizpůsobit velikost zdroje vůči obrazovce, na které bude přehráván. Vykreslování zdroje určeného pro 4K monitor na VGA monitoru přináší zbytečné datové a výkonové nároky. Stejně nešťastné je i řešení vy-

kreslit zdroje pro VGA rozlišení na 4K monitor, kdy nebude moc zřetelné, co je na obrazovce vyobrazeno.

Scénář

Scénář má být dostatečně robustní pro pojmutí veškerých informací k jeho korektivnímu vykreslení. Je nutná definice názvu scénáře stejně jako definice poměru stran, pro který je scénář určen. Ze zkušenosti je výhodné si pamatovat také datum jeho vytvoření a datum poslední změny. Je to výhodné hlavně proto, že je možné rozhodnout, zda scénář uložený na serveru není novější, než scénář, který je nyní přehráván přehrávačem, což vyžaduje synchronizaci.

Scénář musí zaznamenat veškeré zdroje, které bude využívat. Zdroj je v tomto případě url odkaz na obrázek, video či webovou stránku. Také se může jednat o čistý text, který má být vykreslen na obrazovku. Veškeré zdroje musí být opatřeny výchozími hodnotami popisujícími počáteční nastavení. Jedná se o pozici na obrazovce, rozměry, průhlednost, úhel natočení a definice typu písma s barvou v případě čistého textu.

Scénář musí obsahovat i tzv. události. Událost v kontextu scénáře znamená, že pro daný snímek nastala určitá změna. Před vykreslením snímku musí dojít ke zpracování všech událostí, které jsou pro něj určeny. Událost o sobě říká, pro jaký zdroj je určena, pro jaký snímek se má vykonat, jakou vlastnost zdroje mění a definuje funkci, která se provolá pro zjištění nové hodnoty. Tato funkce musí být vytvořena tak, aby nezáleželo na pořadí vykonávání událostí, pokud je definováno více událostí na jeden snímek.

3.2 Uživatelské role

Uživatelská role jsou tři:

- Uživatel
- Editor
- Administrátor

Role uživatel zastřešuje práva samotného ovládání přehrávačů. Editor rozšiřuje pravomoce o možnosti správy a úprav scénářů. Poslední role je schopna spravovat veškeré uživatele i editory systému a přidělovat jim patřičná práva.

3.3 Zabezpečení

Zabezpečení aplikace může konzumovat více scénářů:

- Přihlášení do aplikace pomocí jména a hesla
- Místní síť bez přístupu k internetu
- Server využívající SSL certifikát
- Elektronické podepisování u všech zasílaných zpráv
- Šifrování všech zasílaných zpráv

3.3.1 Přihlášení do aplikace pomocí jména a hesla

Zpřístupnění aplikace pro správu obsahu musí být opatřeno bezpečnostím dostatečně silným heslem pro zamezení přímého přístupu do administrace. Pokud by tomu tak nebylo, může kdokoliv, kdo má přístup k administrační aplikaci, měnit její konfiguraci.

3.3.2 Místní síť bez přístupu k internetu

Prvním řešením je zajistit bezpečnost na síťové vrstvě modelu ISO/OSI. Jde o to, aby veškeré hardwarové komponenty systému komunikovaly po místní síti se zakázaným přístupem k síti internet. Aby mohl potenciální útočník škodit, musí proniknout fyzicky do prostor obchodu. Tam se mu musí podařit napojit na místní síť. Pokud úspěšně provede tyto kroky, lze označit jeho pokus o nabourání za zdařilý. Vnitřní komunikace již šifrovaná není, takže je systém v jeho režii.

V tomto scénáři je nutné zajistit kvalitní distribuci místní sítě. Ve veřejných prostorách nesmí být fyzicky přístupné aktivní síťové prvky, ke kterým by se mohl útočník připojit pomocí UTP kabelu. Pokud je použita bezdrátová síť, musí být šifrována kvalitním algoritmem pomocí silného hesla.

Nevýhodou tohoto scénáře je fakt, že bez přístupu k síti internet nelze sledovat aktuální stav systému a nelze jej ani ovládat. Navíc tímto krokem dojde i k částečnému omezení funkčnosti přehrávání, protože přehrávač nebude moci načítat obsah z internetu – např. webové stránky.

Výhodou pak je bezpečnost řešení s relativně snadnou imlementací a testování funkčnosti řešení. Tento postup je vhodný pouze pro malá řešení bez nutnosti využití internetu.

3.3.3 Server využívající SSL certifikát

Certifikát spolu s veřejným klíčem a certifikační autoritou se mají postarat o vytvoření důvěrného spojení mezi serverem a koncovým klientem. Zajistit má také zašifrování přenášených dat skrze veřejnou síť internet. Korektní funkce je zajištěna pomocí asymetrické šifry. Obě strany vlastní svůj veřejný a soukromý klíč. Pro komunikaci je třeba, aby veřejný klíč byl znám straně druhé. Soukromý klíč nesmí být za žádnou cenu zveřejněn – zvěřejnění by narušilo bezpečnost.

Počátek komunikace probíhá tak, že klient zašle serveru požadavek o spojení. Server odpoví s informacemi o spojení. Klient si ověří u autority korektnost odpovědi a vygeneruje základ šifrovacího klíče, který zašifruje veřejným klíčem serveru. Server rozšifruje pomocí svého klíče základ šífrovacího klíče a obě strany generují hlavní šifrovací klíč. Poté si vzájemně ověří, že budou nadále používat tento nový klíč a komunikace se stává bezpečnou. Až nyní se provede první požadavek, kvůli kterému se prováděli veškeré kroky.

Více informací o této technologii najdete ve zdroji [14].

3.3.4 Elektronické podepisování u všech zasílaných zpráv

Tato metoda se snaží zabezpečit komunikaci na aplikační vrstvě. Data zasílaná sítí jsou zcela veřejná a čitelná, avšak u všech zpráv je přibalen elektronický podpis. Popis funkce elektronického podpisu je k dispozici v kapitole 3.3.5.

Jelikož má elektronický podpis prokázat původ zprávy, musí strana přijímající zprávu znát veřejný klíč strany odesílající. Při přijmutí zprávy provede kontrolu a potvrdí-li se původ, je zpráva důvěryhodná. Nadále již nic nebrání zprávu korektně zpracovat. V opačném případě, když se původ nepotvrdí, zpráva se ignoruje – jedná se o podvrh.

Ačkoliv to na první pohled vypadá jako velmi bezpečný způsob komunikace, je tam jedna drobná trhlina. K tomu, aby přijímající strana mohla ověřit elektronický podpis, musí znát veřejný klíč strany odesílající. Pokud útočník odposlechne tuto komunikaci, může nahradit svým klíčem. Pokud pak všechny zasílané zprávy odchytí, otevře a podepíše svým klíčem, je věrohodný. Aby to nebylo možné, musí systém zajistit prvotní distribuci veřejného klíče. To lze zajistit synchronní šifrou, popř. cizí, veřejnou, důvěryhodnou autoritou.

3.3.5 Elektronický podpis

Elektronický (digitální) podpis (též kvalifikovaný certifikát[15]) je označení pro specifická data, která v elektronické podobě nahrazují klasický vlastnoruční podpis, resp. ověřený podpis.

Elektronický podpis je vytvořen pomocí počítače nad konkrétními daty spolu s identifikací autora podpisu. Pro vytvoření el. podpisu je třeba privátního klíče, který má k dispozici pouze autor podpisu. Druhá strana ověří důvěryhodnost dat pomocí veřejného klíče autora. Správný veřejný klíč autora je kvůli jeho věrohodnosti nutné získat buď přímo od autora či od již důvěryhodné třetí strany. Třetí stranou může být certifikační autorita nebo síť důvěry. Tvorba i ověření elektronického podpisu je soubor matematických operací prováděných nad vstupními daty.

Elektronický podpis umožňuje ověřit několik skutečností:

- Autenticita – zpráva pochází opravdu od správného autora
- Integrita – od vytvoření zprávy nedošlo k její neoprávněné změně
- Nepopiratelnost – autor se nemůže zříct autorství zprávy (k podpisu je třeba privátního klíče, který je spjat pouze se subjektem)
- Časové ukotvení – datum vzniku zprávy může být součástí podpisu

3.3.6 Šifrování všech zasílaných zpráv

Tato metoda funguje na podobném principu jako elektronické podepisování (kapitola 3.3.4) s rozdílem, že se místo připojení podpisu zašifruje celý obsah zprávy. Pokud to někdo zachytí, zpráva je pro něj bez patřičného klíče zcela nečitelná. Problém nastává opět ve chvíli výměny klíčů mezi oběma stranami. Řešení je pak podobné jako v kapitole 3.3.4 o elektronickém podepisování.

Zpráva, která dorazí, je dešifrována. Pokud je korektní, bude zpracována. V opačném případě se ignoruje – s největší pravděpodobností jde o podvrh.

3.3.7 Kombinace scénářů

Nejlepší možností je kombinace scénářů. Pokud je server chráněn pomocí certifikátu SSL, může se prvotní konfigurace šířit od něj a klientské aplikace si mohou ověřit u autority, že jde opravdu o správný zdroj. Tímto postupem lze obohatit jak elektronický podpis, tak i šifrování zpráv. Toto celé v kombinaci se zabezpečením místní sítě zvyšuje odolnost vůči napadení.

4 Analýza a návrh

Na základě předchozí kapitoly 3 jsou navrženy požadavky na systém. Popis možného použití systému popisují případy užití za pomoci scénářů. Součástí této kapitoly je i analytický návrh strutury aplikace.

4.1 Požadavky na systém

4.1.1 Funkční požadavky

- systém umožní zobrazit obrázek (obecně obsah)
- systém umožní přehrát video (obecně obsah)
- systém umožní zobrazit text (obecně obsah)
- systém umožní nastavit font textu
- systém umožní nastavit velikost textu
- systém umožní nastavit pozici textu na obrazovce
- systém umožní nastavit pozici obrázku na obrazovce
- systém umožní nastavit velikost obrázku na obrazovce
- systém umožní nastavit pozici videa na obrazovce
- systém umožní nastavit velikost videa na obrazovce
- systém umožní nastavit, jak dlouho bude obsah zobrazen
- systém umožní vytvoření scénáře se shlukováním více obsahu do jednoho
- systém dokáže zobrazit scénář na jedné obrazovce
- systém umožní distribuci jednotlivých scénářů mezi přehrávače
- systém musí vytvořit scénář s relativní velikostí
- systém musí vytvořit scénář vázaný na poměr stran obrazovky

- systém umožní nastavit počáteční a koncovou animaci pro obsah
- systém dokáže informovat o stavu jednotlivých fyzických částí systému
- systém předává informace mezi jednotlivými fyzickými částmi v reálném čase
- systém umožní vypsání aktuálních aktivit přehrávače
- systém dokáže spolupracovat s jakýmkoliv zobrazovacím zařízením, které obsahuje HDMI port
- systém umožní vyrovnat úhlové natočení zobrazovacího zařízení
- systémové fyzické části komunikují pomocí síťového rozhraní TCP/IP
- systémová komunikace fyzických částí je dostatečně imunní proti vložení cizí informace nepovolanou osobou
- systém obsahuje centrální prvek
- centrální prvek eviduje veškeré klíčové prováděné akce
- centrální prvek ukládá veškeré klíčové prováděné akce
- systém umožní upload zdrojových souborů
- systém umožní správu zdrojových souborů
- systém umožní zdrojové soubory přejmenovat
- systém umožní správu scénářů
- systém umožní správu a konfiguraci přehrávačů

4.1.2 Výkonnostní požadavky

- přehrávaný scénář musí být přehráván plynule
- editor scénáře musí být přístupný na všech počítačích s běžným operačním systémem obsahujících hardwarově akcelerovanou grafickou nástavbu
- přehrávač scénáře musí být dostupný na všech počítačích s běžným operačním systémem obsahujících hardwarově akcelerovanou grafickou nástavbu

4.1.3 Designové požadavky

- ovládání systému je pokud možno intuitivní
- přehrávaný scénář je zobrazen přes celou obrazovku

4.2 Případy užití

Model případů užití na obrázku 4.1 popisuje základní operace, které mohou aktéři provést. Použití aktéři:

- uživatel
- editor
- administrátor

Nejnižší pravomoce má role uživatel. Je omezen prakticky pouze na vzdálené ovládání přehrávače. Vyšší rolí je tzv. editor, který může v systému měnit scénáře. Může i spravovat zdrojové soubory pro scénáře. Nejvyšší rolí je administrátor – smí určovat další osoby, které mohou do systému.

4.2.1 Scénáře případů užití

UC_01_Přidat uživatele

1. administrátor klikne na tlačítko "přidat uživatele"
2. systém zobrazí patřičný formulář
3. administrátor vyplní e-mail, heslo, roli a klikne na tlačítko uložit
4. systém zvaliduje vstupní data
5. vyskytne-li se při validaci chyba, pokračuje se bodem 3, jinak systém uživatele uloží

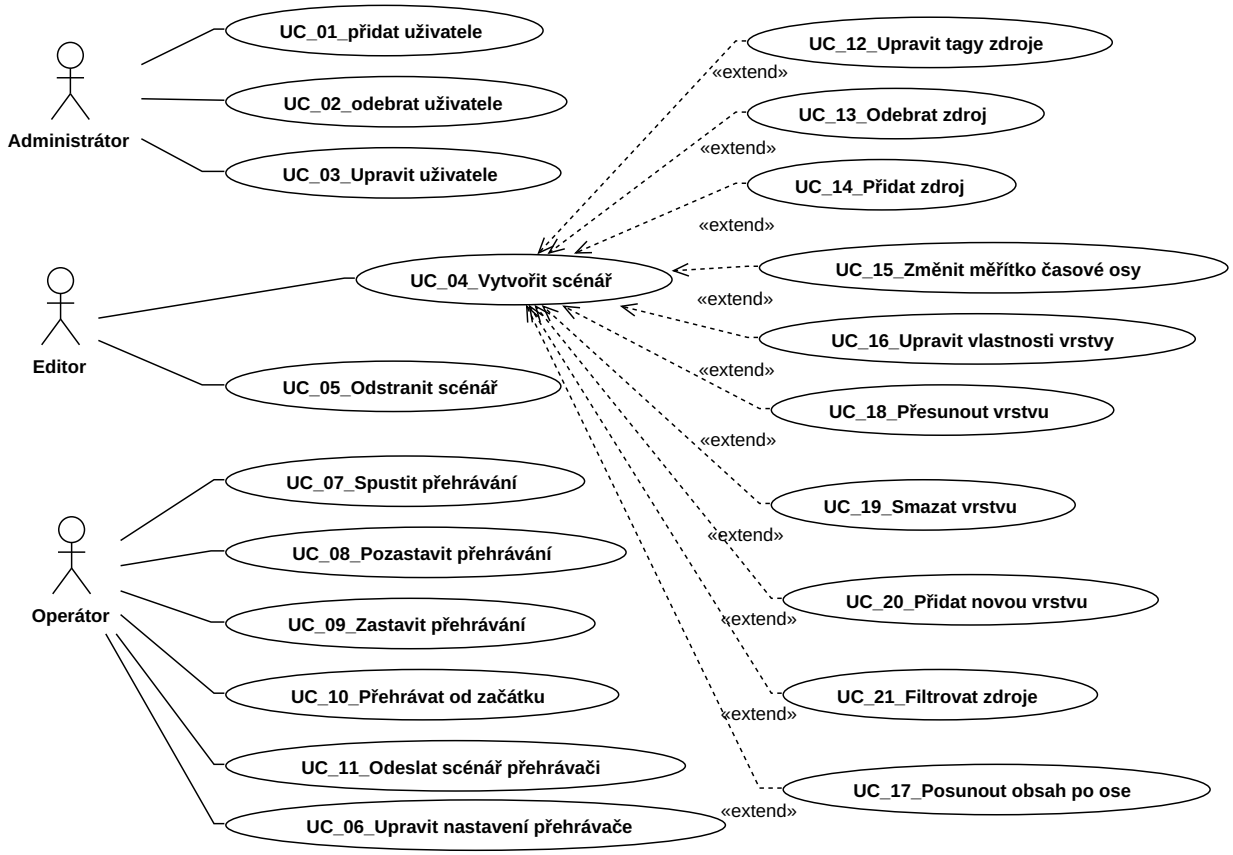
UC_02_Odebrat uživatele

1. administrátor klikne na tlačítko "odebrat uživatele"
2. systém zobrazí dialog s dotazem, zda chce administrátor opravdu smazat uživatele
3. pokud administrátor potvrdí smazání, systém uživatele odstraní. V opačném případě se mazání ignoruje.

UC_03_Upravit uživatele

1. administrátor klikne na tlačítko "upravit uživatele"
2. systém zobrazí patřičný formulář s předvyplněnými údaji o upravovaném uživateli

3. administrátor upraví e-mail, heslo, roli a klikne na tlačítko uložit
4. systém zvaliduje vstupní data
5. vyskytne-li se při validaci chyba, pokračuje se bodem 3, jinak systém uživatele uloží



Obrázek 4.1: Model případů užitíl (zdroj: autor)

UC_04_Vytvořit scénář

1. editor klikne na tlačítko "vytvořit scénář"
2. systém zobrazí editor scénáře
3. systém dále čeká na provedení činnosti uživatelem – danou činnost deleguje na příslušný sub-systém (např. přidání nové vrstvy, změna měřítka časové osy atd.)
4. pokud předchozí činnost není "ukončit", pokračuje se bodem 3.

UC_05_Odstranit scénář

1. editor klikne na tlačítko "odebrat scénář"

2. systém zobrazí dialog s dotazem, zda chce editor opravdu smazat scénář
3. editor potvrdí smazání
4. pokud editor potvrdí smazání, systém scénář odstraní

UC_06_Upravit nastavení přehrávače

1. operátor klikne na tlačítko "upravit nastavení přehrávače"
2. systém zobrazí editační formuláře pro přehrávač
3. operátor upraví název či úhel natočení zobrazovacího zařízení
4. operátor klikne na tlačítku "uložit"
5. systém provede kontrolu správnosti vyplnění
6. pokud je formulář v pořádku, systém data odešle na synchronizační server. Jinak se pokračuje bodem 3.

UC_07_Spustit přehrávání

1. operátor klikne na tlačítko "play"
2. systém odešle informaci na synchronizační server

UC_08_Pozastavit přehrávání

1. operátor klikne na tlačítko "pause"
2. systém odešle informaci na synchronizační server

UC_09_Zastavit přehrávání

1. operátor klikne na tlačítko "stop"
2. systém odešle informaci na synchronizační server

UC_10_Přehrávat od začátku

1. operátor klikne na tlačítko "reset"
2. systém odešle informaci na synchronizační server

UC_11_Odeslat scénář přehrávači

1. operátor klikne na tlačítko "upravit nastavení přehrávače"
2. systém zobrazí seznam všech dostupných scénářů
3. operátor vybere ze seznamu jeden scénář
4. systém odešle informaci na synchronizační server

UC_12_Upravit tagy zdroje

1. editor klikne na "upravit tagy zdroje"
2. systém zobrazí formulář s výčtem tagů
3. editor upraví tagy
4. editor deaktivuje formulář
5. systém tagy k danému zdroji uloží

UC_13_Odebrat zdroj

1. editor klikne na "odebrat zdroj"
2. systém zobrazí dialog s dotazem na vykonání akce
3. pokud editor potvrdí, systém zdroj smaže

UC_14_Přidat zdroj

1. editor klikne na "nahrát nový zdroj"
2. systém zobrazí formulář pro nahrátí nových souborů
3. editor vybere nové soubory
4. editor zmáčkne nahrát
5. systém přijme soubory
6. systém obnoví seznam zdrojů

UC_15_Změnit měřítko časové osy

1. editor klikne na tlačítko "zvětšit (resp. zmenšit) měřítko osy"
2. systém přepočítá měřítko časové osy patřičným směrem

UC_16_Upravit vlastnosti vrstvy

1. editor klikne na vrstvu
2. systém vizuálně znázorní zaktivování vrstvy
3. systém načte editovatelná data o vrstvě do editoru vlastností
4. editor upraví v editoru vlastností jednotlivé vlastnosti
5. systém uloží vlastnosti při jakékoliv jejich korektní změně

UC_17_Posunout obsah po ose

1. editor uchopí myší grafický prvek určený pro změnu umístění vrstvy v čase
2. systém vizuálně znázorní uchopení vrstvy
3. editor umístí kurzor s uchopenou vrstvou na nové místo a upustí vrstvu
4. systém umístí vrstvu na dané místo
5. systém informuje náhledový přehrávač o změně playlistu

UC_18_Přesunout vrstvu

1. editor uchopí myší místo pro změnu pořadí vrstvy
2. systém vizuálně znázorní uvolnění ze seznamu
3. editor umístí kurzor s uchopenou vrstvou na nové místo a upustí vrstvu
4. systém umístí vrstvu na dané místo
5. systém informuje náhledový přehrávač o změně playlistu

UC_19_Smazat vrstvu

1. editor spustí událost pro odebrání vrstvy

2. systém odebere vrstvu ze seznamu
3. systém informuje náhledový přehrávač o změně playlistu

UC_20_Přidat novou vrstvu

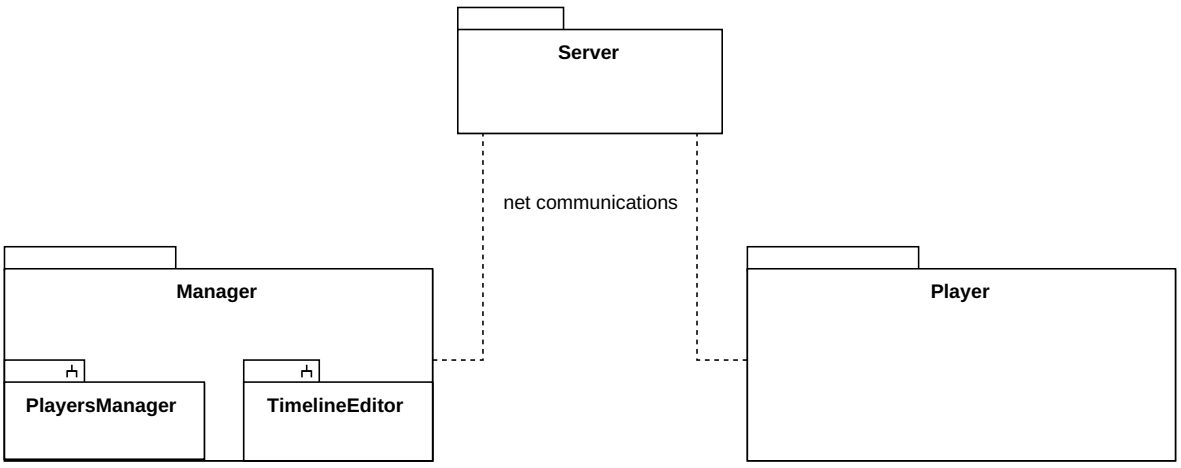
1. editor spustí událost pro přidání zdroje
2. systém přidá do seznamu novou vrstvu na nejspodnější místo
3. systém vrstvu naplní meta daty
4. systém informuje náhledový přehrávač o změně playlistu

UC_21_Filtrovat zdroje

1. editor vyplní do příslušného textboxu část/celý název zdroje či jeho tag
2. systém zobrazí seznam odpovídajících zdrojů

4.3 Struktura systému

Návrh realizace aplikace je rozdělit ji na 3 samostatné aplikace, kdy každá bude obstarávat své specifické funkce. Na obrázku 4.2 je návrh jednotlivých balíčků a to synchronizační-datový server, správce přehrávačů s editorem scénáře a přehrávač scénáře. Navrhovaná komunikace těchto částí je pomocí datové šítě.

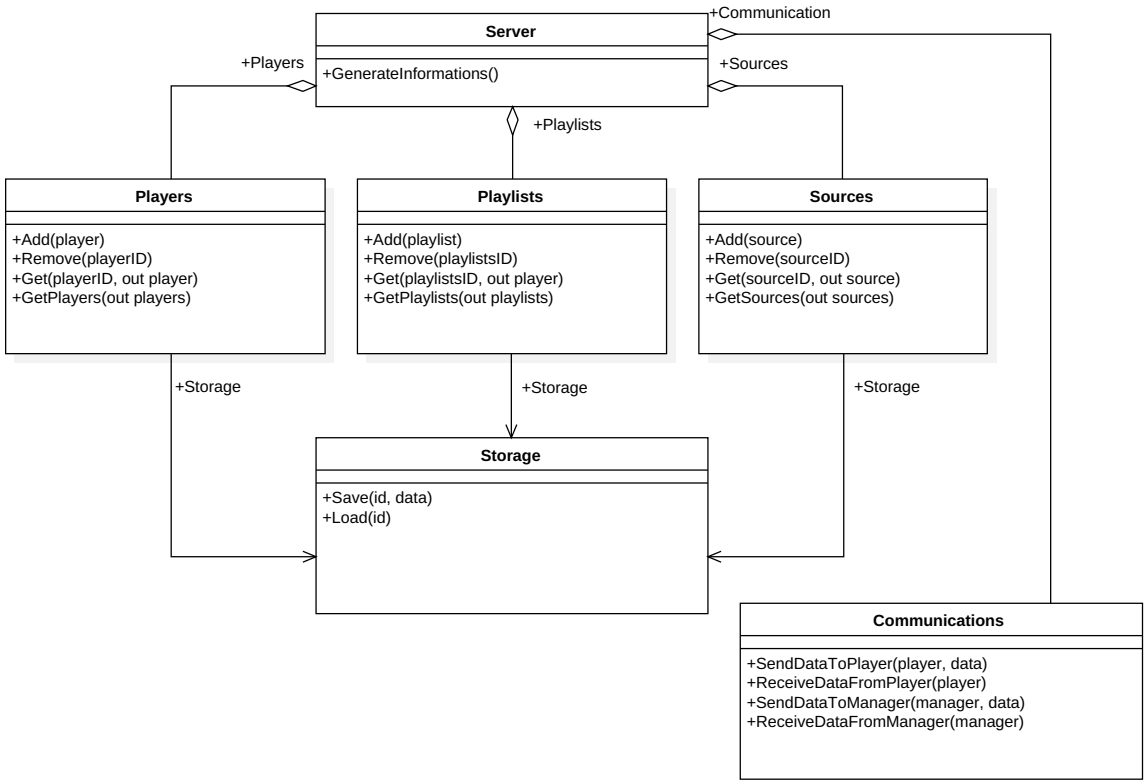


Obrázek 4.2: Diagram balíčků (zdroj: autor)

4.3.1 Analytický model serveru

Diagram 4.3 popisuje návrh serverové části, jejímž obsahem je kromě výchozí spustitelné aplikace i komunikační modul. Skrze tento modul prochází veškerá síťová komunikace. Dále jsou obsaženy moduly:

- zpracování přehrávačů – ukládání, načítání a funkcionalita pro ovládání
- zpracování scénářů – ukládání, načítání a distribuce přehrávačům
- změnový observer – generování informací o změnách při připojení, resp. odpojení přehrávače (manageru) atd.
- zpracování zdrojů – upload nových zdrojů, jejich správa a poskytnutí zdrojů zvenku



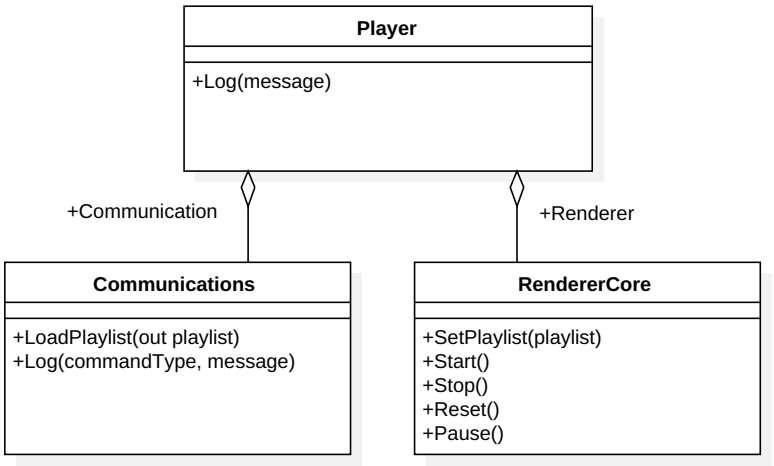
Obrázek 4.3: Analytický diagram – server (zdroj: autor)

4.3.2 Analytický model přehrávače

Diagram 4.4 popisuje strukturu přehrávače médií. Interně obsahuje komunikační modul pro podporu komunikace s okolím a jádro pro vykreslování (renderování) obrazu. Vykreslovací jádro v sobě musí mít zapouzdřenou funkčnost:

- inicializace nového playlistu
- spuštění přehrávání
- pozastavení přehrávání
- stopnutí přehrávání

Komunikační modul je schopen nepřímo volat metody renderujícího jádra. Nepřímo znamená, že on sám o cílovém objektu povědomí nemá. Komunikace mezi těmito dvěma členy probíhá pomocí hlavní aplikace *player*, která se postará o propojení příslušných příkazů.



Obrázek 4.4: Analytický diagram – player (zdroj: autor)

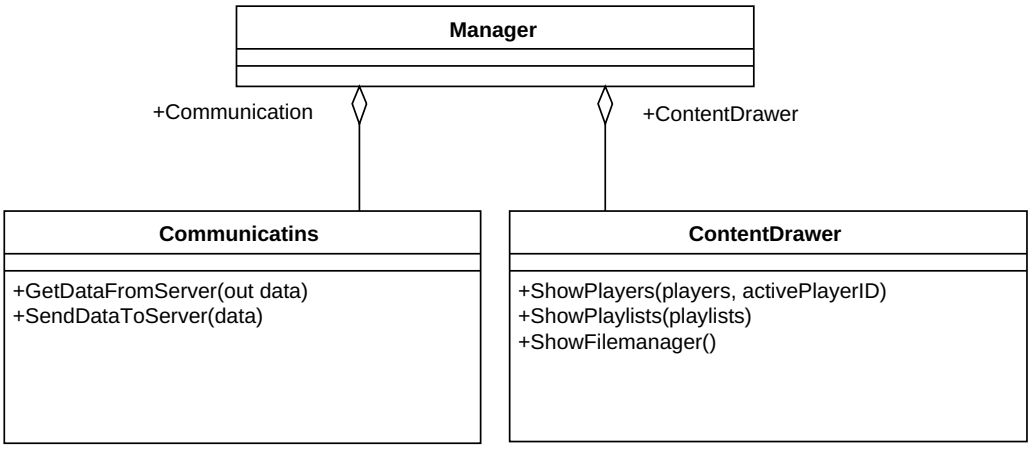
4.3.3 Analytický model správce

Diagram 4.5 rozděluje aplikaci správy přehrávačů do určité hierarchie. Pro správnou funkčnost je zahrnut komunikační modul. Celá aplikace má *ContentController*, který rozhoduje o tom, jaká část aplikace bude zobrazena. Aplikace je rozdělena na části příhlášení, správy přehrávačů, správy scénářů a editoru scénáře. Editor scénaře je z důvodu zvýšení přehlednosti podrobněji vyobrazen v diagramu 4.6.

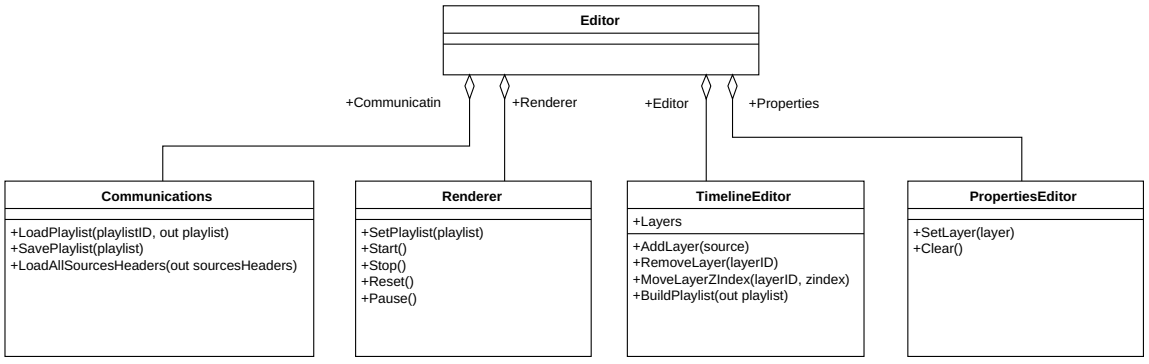
Data pro jednotlivé části systému zajišťuje obousměrný komunikační modul.

PlaylistBuilder má být schopen převést scénář na konfiguraci grafických prvků pro reprezentaci uživateli a obráceně – vytvořit playlist z grafických prvků.

Dalším zajímavým prvkem je *ZoomSupport*, jelikož celá časová osa musí být schopná přiblížení, resp. oddálení – tzn. zvýšení, resp. snížení rozpětí osy a s tím spojeného přepočítání rozměrů veškerých vrstev.



Obrázek 4.5: Analytický diagram – manager (zdroj: autor)



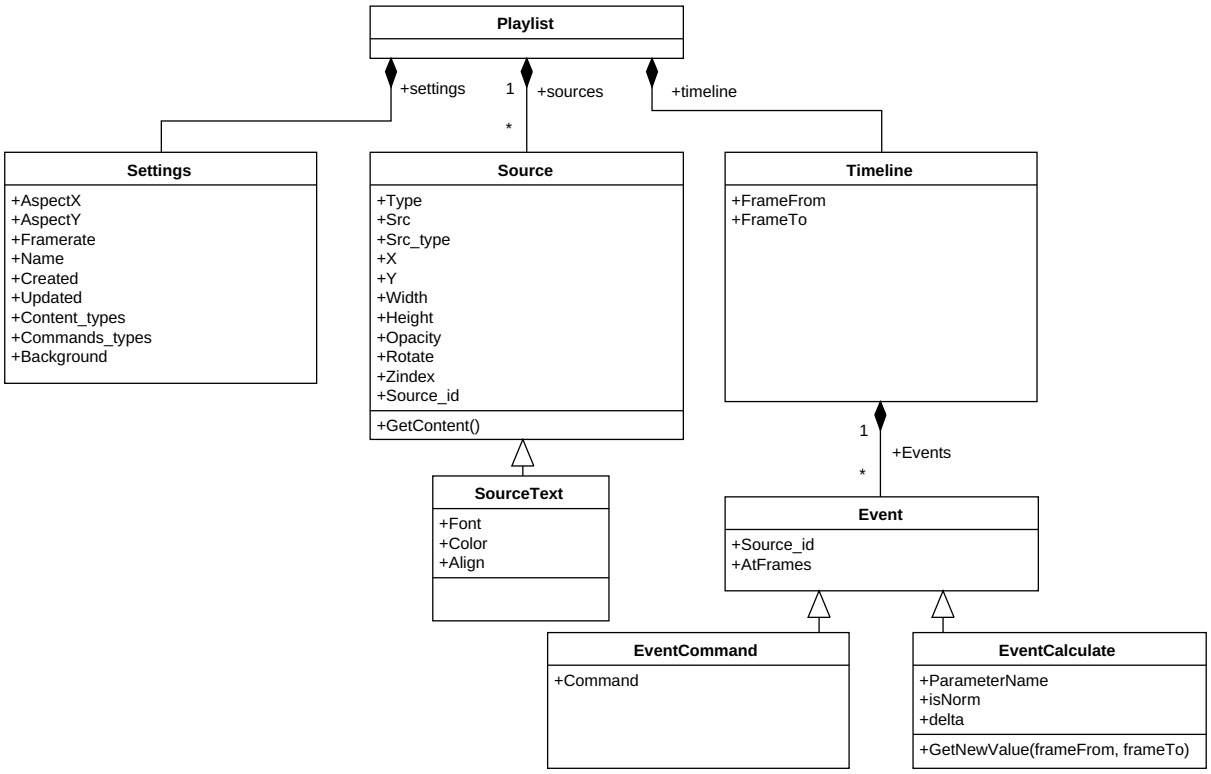
Obrázek 4.6: Analytický diagram – editor (zdroj: autor)

4.4 Vnitřní struktura scénáře

Objektový návrh scénáře se v sobě snaží zapouzdřit funkčnost navrženou v kapitole 3.1.4. Scénář se skládá z obecného nastavení, odkazů na zdrojové soubory a popisu jednotlivých událostí.

Nenápadné, avšak velmi podstatné, jsou dvě funkce navržené v diagramu 4.7. První v rámci třídy *Source* s názvem *GetContent()* má za úkol vrátit odkaz na zdroj v případě video, audio či obrazového zdroje a samotný text v případě textového zdroje.

Zajímavější funkcí je *GetNewValue()* ve třídě *EventCalculate(frameFrom, frameTo)*. V ní je zapouzdřena funkce pro přepočet určité vlastnosti. Z parametrů funkce se dozví, z jakého snímku vychází a jaký je potřeba spočítat. V rámci své třídy zná jméno parametru (*parameterName*), který má hledat a načíst jeho hodnotu. Poté má funkce veškeré potřebné údaje a návratovou hodnotou je nová hodnota pro zjišťovaný parametr. Tato funkce se volá v každém snímku, pro který je určena. Přepočet může probíhat lineárně, pomocí křívek vyšších řádů nebo zcela náhodně. Pomocí parametru *parameterName* lze tak nastavit jakoukoliv vlastnost nesenou třídou *Source* a spočítat hodnotu novou. Tato metoda se postará o animace a posuny. V této funkci lze zapouzdřit i určité skriptování uživatelem, čímž si může vytvořit vlastní animaci.



Obrázek 4.7: Analytický diagram – playlist (zdroj: autor)

5 Implementace

Tato část je zaměřena na samotnou implementaci systému. Jsou zde popsány technologie, které byly použity k tvorbě systému. Dále popisuje implementaci klíčových částí systému. Zde je nutno podotknout, že nebyla provedena implementace všech požadavků definovaných v kapitole 4. Implementace se omezila na problematiku spojenou s renderováním obsahu (scénáře) v reálném čase, správu přehrávačů a správu scénářů s editorem scénáře. Nejsou tak např. implementovány případy užití, které řeší uživatelské účty. Celý systém je tak postaven nad jediným účtem, který je pro ukázku funkčnosti plně dostačující. Dále byla vynechána problematiky bezpečnosti, která je sice velmi důležitým hlediskem, nicméně není předmětem této diplomové práce.

Na základě diagramu 4.2 byl systém rozdělen na tři samostatné části. Samostatným částem jsou věnovány kapitoly 5.4 (server), 5.5 (správce) a 5.6 (přehrávač). Kapitola 5.3 popisuje implementaci komunikace mezi jednotlivými částmi systému.

5.1 Použité technologie

S ohledem na vzrůstající oblibu webových aplikací byla tato práce směřována především na webové technologie. Díky tomuto kroku bude možno aplikaci spustit bez větších obtíží na více různých platformách a to bez nutnosti vývoje nové aplikace pro danou platformu.

Pro serverovou část je volena relativně nová technologie – Node.js. Tento systém nabízí kompatibilitu s předními operačními systémy, proto je instalace velmi snadná – v podstatě se jedná o stažení a nainstalování jednoho softwarového balíčku.

Naopak klienská část je tvořena webovými technologiemi ve specifikaci HTML5, CSS a JavaScript. Více o těchto technologiích v sekcích 5.1.2, 5.1.3 a 5.1.4.

Pro síťové propojení klientské a serverové strany v reálném čase byla využita knihovna socket.io (více o této knihovně v sekci 5.1.6).

5.1.1 Technologie Node.js

Node.js [18] je softwarový systém navržený pro psaní vysoce škálovatelných internetových aplikací, především webových serverů. Programy pro Node.js jsou psané v jazyce JavaScript (viz. kapitola 5.1.4) hojně využívajícím model událostí a asynchronní I/O operace pro minimalizaci režie procesoru a maximalizaci výkonu.

Klíčový rok pro Node.js je rok 2009, kdy byla vytvořena jeho první veřejná verze. V době psání této práce implementuje *V8 JavaScript engine* od společnosti Google spolu s několika standardními knihovnami. Technologie Node.js zajišťuje možnost spustit JavaScript kód nikoliv v prohlížeči, ale jako samostatnou aplikaci.

Technologie Node.js podporuje balíčkovací systém *npm* [26]. Balíčkovací systém umožňuje spravovat používané cizí části kódu. Je to výhodné z pohledu zajištění konzistence verzí jednotlivých balíčků a také kvůli odbourání nutnosti vlastní zdroje obohatit o potřebné balíčky třetích stran – výrazné ušetření místa v uchování aplikací. Projekt si tak sebou nese pouze reference na jednotlivé balíčky a informaci o potřebných verzích. Při vysazení na produkční prostředí dojde jednoduchým příkazem ke stažení potřebných balíčků.

Express – webový framework

Express [21] je označení jednoho z frameworků určených pro technologii Node.js. Framework umožňuje deklaraci s nastavením url cest pro *REST API* , resp. webový server – podporuje *path rewriting*. Vytvoření ukázkové aplikace, která bude naslouchat na určitém portu s definicí cest, je uvedena v ukázce 5.1. Uvedení několika ukázkových řádků kódu zajistí naslouchání na adresách GET / a POST / s odesláním odpovědi na dotaz. Jako odpověď lze odeslat čistý text, formátovaný text dle určitého standardu (HTML, XML, JSON apod.) nebo konkrétní binární soubor. V dané odpovědi se nastaví hlavička tak, aby byl klient informován o typu dat v těle odpovědi. Tímto je řešen výše uvedený *path rewriting*. Pod určitou url je schován soubor, který se fyzicky nachází na jiné adrese. Lze zahrnout ověření uživatele a rozhodnout tak o právech pro přístup k danému souboru.

Správce souborů

Aby editor scénáře mohl korektně fungovat, je zapotřebí zahrnout způsob správy zdrojových souborů. Jelikož je v dnešní době správce souborů běžnou aplikací, bylo vybráno jedno z konečných řešení vytvořených jiným vývojářem a nabízených pod neomezující licencí. Volen ým řešením je *Cloud CMD* [22] distribuovaný jako klasický balíček technologie Node.js.

Listing 5.1: Vytvoření REST API pomocí knihovny Express (jazyk JavaScript)

```
// pripojeni zdrojovych kodu knihovny
var express = require('express');
// vytvoreni instance serveru
var app = express();

// vytvoreni REST API
// metoda GET
app.get('/', function(req, res, next) {
  res.send('GET Hello World');
});

// metoda POST
app.post('/', function (req, res, next) {
   res.send('POST Hello World');
});

// spusteni naslouchani na portu 3000
app.listen(3000, function () {
   console.log('Naslouchám na portu 3000!');
});
```

5.1.2 Značkovací jazyk HTML

HTML je název značkovacího jazyka užívaného pro tvorbu webových stránek (v systému World Wide Web), které jsou propojeny hypertextovými odkazy. HTML ve verzi 5 přináší řadu specifikací a vylepšení hlavně v oblasti dynamických aplikací. Výhodou HTML5 je i to, že se jedná o rozšíření staršího, velmi užívaného standardu HTML4. Z pohledu vývoje aplikace pod HTML4 přechod na verzi vyšší prakticky neexistuje. Aplikaci stačí rozšířit o nové, aplikací vyžadované prvky z HTML5.

Popisovaná praktická část využívá ze specifikace HTML5 hlavně následující prvky. V prvé řadě jsou to prvky pro práci s multimediálním obsahem. Pomocí těchto prvků je vyřešeno zpracování audio a video souborů. Toto v dřívějších specifikacích chybělo, a tak se musely využívat řešení třetích stran. Přínosem tohoto přístupu je, že stačí znát cestu k příslušnému zdroji spolu s informací, o jaký typ zdroje se jedná. O následné dekódovaní se postará webový prohlížeč. Dalším využitým velmi mocným prvkem z HTML5 je tzv. *canvas*. Canvas je prvkem zajišťujícím možnosti práce s 2D i 3D prostorem přímo ve webovém prohlížeči. Umožňuje vykreslovat základní grafická primitiva jako body, čáry, trojúhelníky apod. Dále umožňuje vykreslit obrázky a text. Podporuje barevné a geometrické transformace vykreslovaných prvků. K vykreslení využívá akceleraci pomocí grafické karty počítače. Více informací o problematice HTML5 je dostupných na [23].

5.1.3 Kaskádové styly (CSS)

CSS popisuje způsob zobrazení elementů na stránkách napsaných v jazycích HTML, XHTML nebo XML. Jedná se o soubor pravidel, pomocí nichž dochází k ovlivnění vykreslování jednotlivých prvků. Pomocí definice CSS pravidel lze grafickému rozhraní aplikace propůjčit jedinečný vzhled. Lze definovat více pravidel pro řešení totožné věci, ovšem jiným způsobem. Tím se docílí změny vzhledu záměnou seznamu pravidel. Jeden seznam pravidel je pak nazýván tzv. *layout*. CSS lze mapovat na samotné prvky ze specifikace HTML, na identifikátory použité v rámci prvků HTML či obecné třídy přiřazené k HTML prvkům. Grafická pravidla se zapisují buď přímo do webové HTML stránky, nebo se připojí pomocí externího souboru – to je lepší a čistější metoda, protože připojením externího souboru dojde k oddělení rozložení prvků samotné aplikace a definice grafických pravidel. Více o této technologii je k dispozici na [24].

5.1.4 Programovací jazyk JavaScript

Jedná se o objektově orientovaný skriptovací jazyk používaný hlavně ve webových technologiích jak na klientské straně, tak i na serverové.

1. interpretovaný – bez nutnosti kompilace
2. objektový – využívá objektů prohlížeče a zabudovaných objektů
3. závislý na prohlížeči – podpora většiny prohlížečů
4. case sensitive – záleží na velikosti písem v zápisu
5. syntaxí podobný jazykům C, Java aj.
6. dynamicky typovaný

Základní datové typy podporované tímto jazykem jsou *number* pro zastoupení čísla, *string* pro textový řetězec, *boolean* pro pravdivostní hodnotu a *object* pro vše ostatní. I když je to jazyk objektově orientovaný, neobsahuje pojem *třída* známý z jiných objektových jazyků. Object je ve své podstatě slovník mapující obsažené proměnné na jejich názvy. Jedná se o kontejner. JavaScript implementuje i určitá paradigma z funkcionálního programování a dovoluje uložit funkci jako běžný object. Toto paradigma dovolí tvorbu lambda výrazů, což dovolí předat funkci v parametru jiné funkce. V častých případech se jedná o tzv. *callback*. Volaná funkce nevrátí výsledek pomocí synchronního klíčového slova *return*, ale pomocí zavolání funkce předané v parametru. Paradigma také dovoluje funkci využít jako konstruktor objektu. Protože zde funguje prototypová dědičnost, object může být předlohou jiného objectu. Jazyk tak specifikuje nový pohled na OOP.

Více o tomto jazyku je k dispozici na [3].

5.1.5 Knihovna jQuery

jQuery [4] je JavaScriptová knihovna podporující širokou škálu webových prohlížečů. Zaměřuje se hlavně na interakci mezi HTML a JavaScriptem. Obsahuje rozsáhlou implementaci pro práci s HTML strukturou (tzv. HTML DOM – struktura HTML dokumentu). Pomocí jQuery lze vytvářet, mazat, upravovat i vyhledávat jednotlivé elementy dle specifických klíčů.

5.1.6 Knihovna Socket.IO

Socket.IO [25] je knihovna postavená nad jazykem JavaScript. Knihovna přináší metody pro oboustrannou komunikaci v reálném čase pomocí webových socketů. Odstraňuje tak nedostatek webových aplikací komunikujících pomocí protokolu HTTP, kdy komunikace probíhá vždy na základě iniciativy klienta, tzn. klient vznese dotaz a server odpoví. Socket.IO umožňuje vyvolat určitou událost u konkrétního klienta na přání serveru. Vyvolanou událost lze obohatit i o data, na která následně může klient reagovat. Pomocí této knihovny lze tak relativně snadno docílit dynamických webových aplikací pracujících bez klasického obnovení stránky pro načtení nových dat. Data se načítají asynchronně na pozadí a jsou vykreslována v reálném čase do grafického rozhraní.

Příkladem obrácené komunikace server-klient je realizace internetového chatu. Klient odešle zprávu na server a uvede jednoho příjemce. Server vyhledá příjemce a vyvolá u příslušného klienta událost pro oznámení nové zprávy s datovou složkou určující odesílatele a tělo zprávy. Bez obnovení webového prohlížeče je tak docíleno minimálního zpoždění mezi odesláním a přijetím zprávy (samozřejmě v závislosti na síťové lince a vytížení serveru).

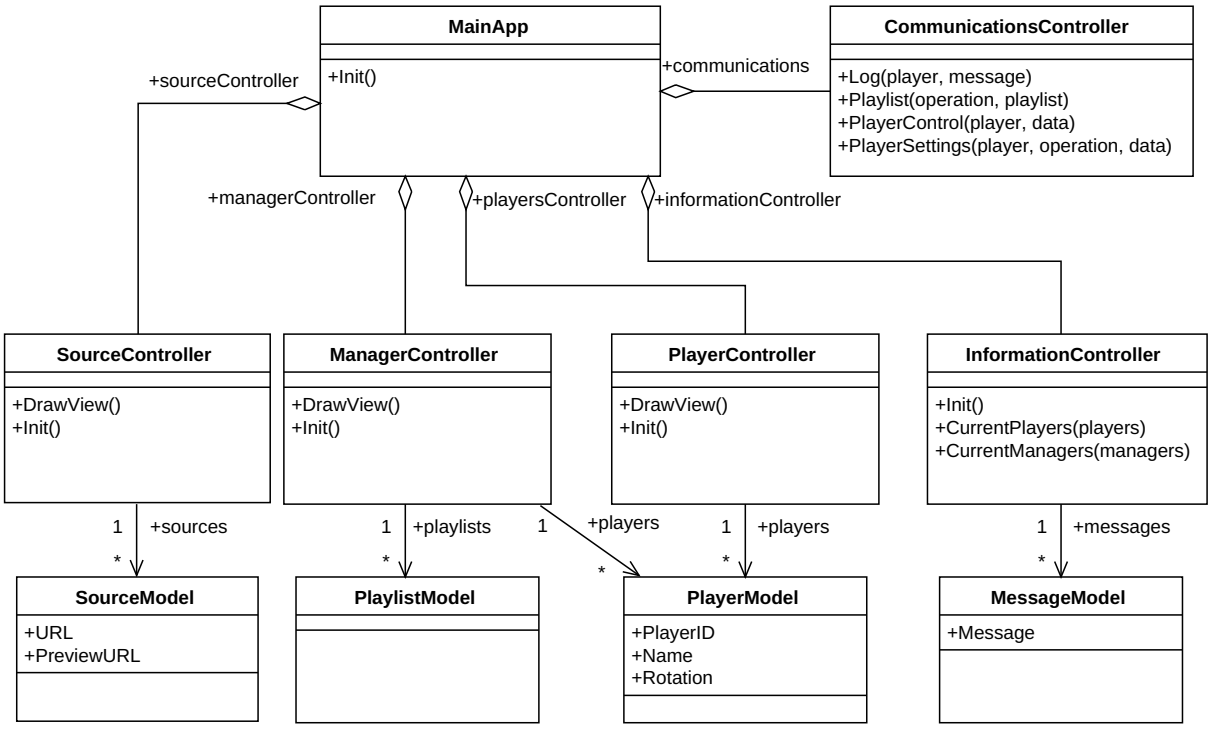
Protože klient i server je implementován pomocí jazyku JavaScript, lze knihovnu využít na obou stranách.

5.2 Diagram tříd

Diagram tříd popisuje reálnou implementaci systému. Snaží se zachytit aktuální obraz struktury aplikace. Veškeré použité diagramy tříd jsou dostupné i na přiloženém CD.

5.2.1 Server

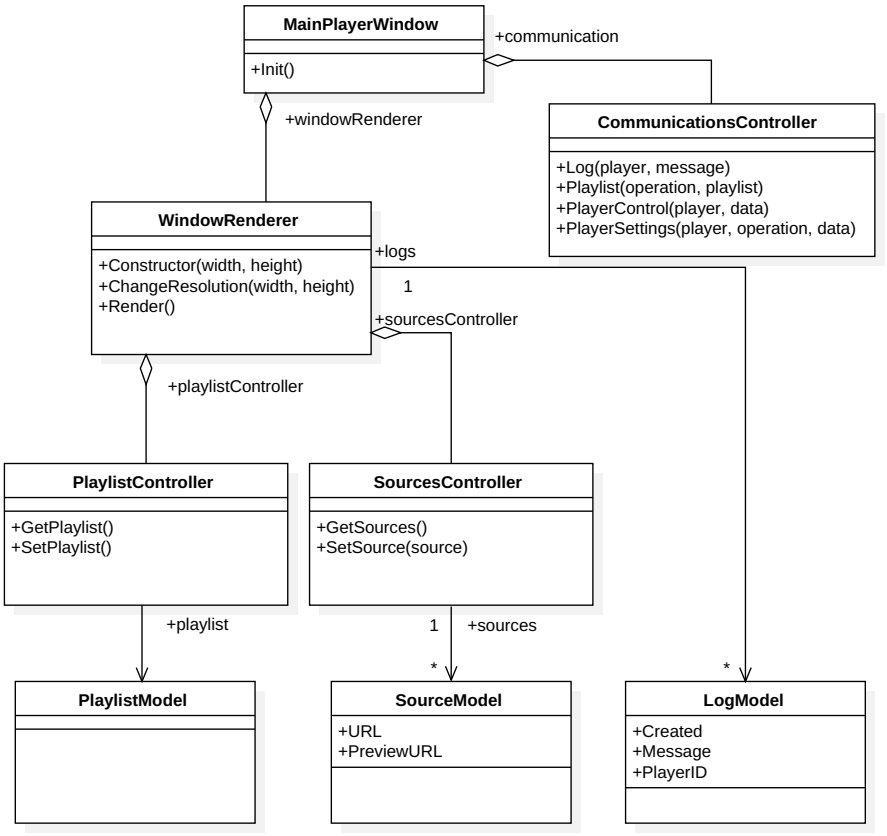
Diagram 5.1 popisuje rozložení tříd aplikace serveru. Spouštěcím bodem aplikace je třída *MainApp*. Tato hlavní třída má k dispozici sadu controllerů, které mají vždy za úkol zapouzdřit určitou funkcionalitu. *SourceController* zpřístupňuje práci se zdroji. Jelikož se jedná o webovou aplikaci, jejím cílem je vykreslit view zpřístupňující uživateli správu zdrojů. V podobném schématu pracují i třídy *ManagerController* a *PlayerController*. Všechny tyto tři controllery umožňují přístup k aplikacím, které jsou popsány v dalších kapitolách. *CommunicationController* zastává funkci komunikace pomocí signal.io knihovny. Klient si stáhne danou aplikaci pomocí jednoho z předchozích controllerů a další komunikace probíhá skrz tento controller. Posledním controllerem je *InformationController*. Jeho funkcí je informovat o změnách v připojených přehrávačích a správcích. V případě, že dojde k připojení nového přehrávače, má za úkol vygenerovat událost, pomocí které se informují o změně připojení správci. Třída *PlaylistModel* je popsaná v kapitole 4.4.



Obrázek 5.1: Class diagram – server (zdroj: autor)

5.2.2 Přehrávač

Spouštěcí místo klientské aplikace popsané v 5.2 tvoří třída *MainPlayerWindow*. K ní se přidávají třídy zajišťující komunikaci (*CommunicationController*) a zajišťující práci s vykreslováním scénáře (*WindowRenderer*). Třída WindowRenderer obsahuje renderující smyčku s korekturou snímkové frekvence. Pro správné vykreslení potřebuje dekodér scénáře a funkcionality, které dokáží ze scénáře převzít zdroje. WindowRenderer generuje logovací zprávy, které se pak pomocí komunikační třídy posílají na server. Třída *PlaylistModel* je popsaná v kapitole 4.4.



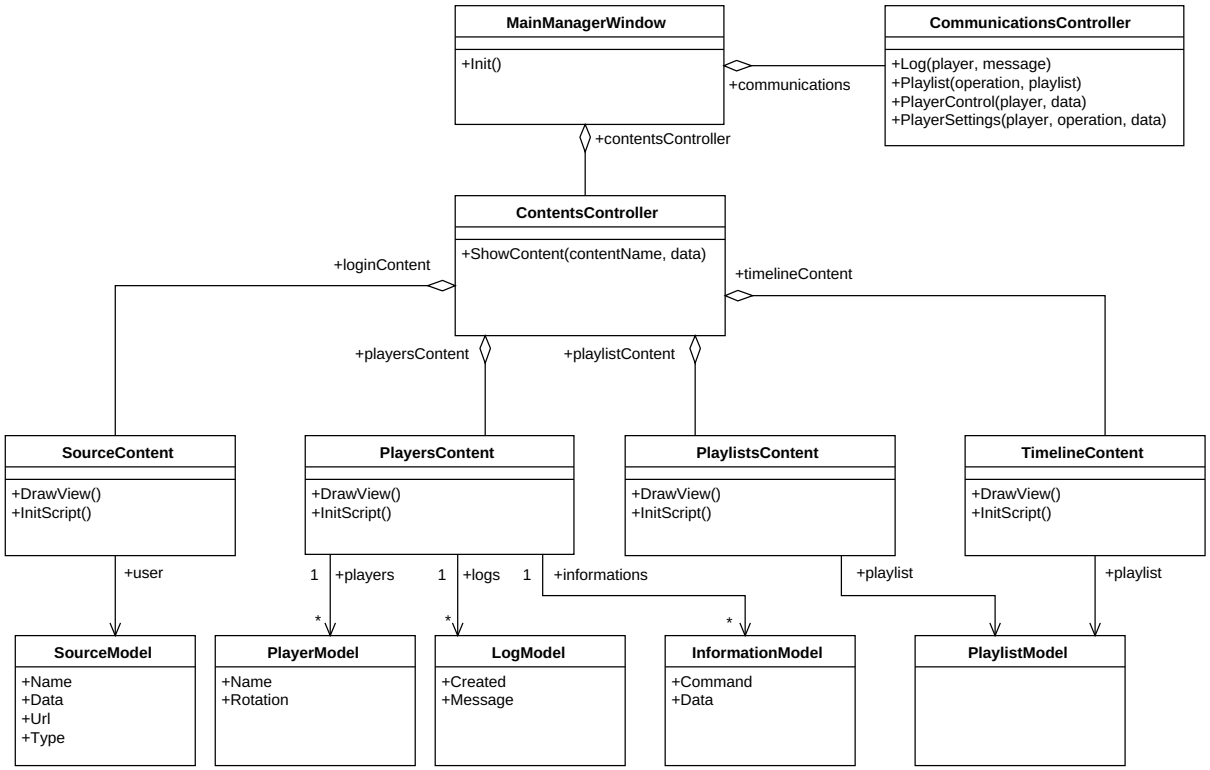
Obrázek 5.2: Class diagram – player (zdroj: autor)

5.2.3 Správce

Protože aplikace správce je tvořena více technologickými vzájemně odlišenými funcionalitami, je rozdělena do více diagramů. Nejprve diagram 5.3 znázorňuje celkový pohled vyjma tvorby scénáře. Tvorba scénáře je v samostatném diagramu 5.4.

Celkový pohled

Aplikace správce vychází opět z jedné inicializační třídy – *MainManagerWindow*. Aplikace je určena pro běh u klienta a je rozdělena do jednotlivých oddílů. V jeden čas může být vykreslen pouze jeden oddíl. O tuto závislost se stará třída *ContentController*. Třída zajistí, že je vykreslen právě jeden oddíl z následujícího výčtu: *PlayersContent*, *PlaylistContent* a *TimelineContent*. Pomocí PlayersContent se grafickému rozhraní zpřístupní správa přehrávačů včetně možnosti ovládání a odeslání scénáře konkrétnímu přehrávači. PlaylistContent zajišťuje správu scénářů. O samotnou úpravu scénáře se postará TimelineContent. Ten je ovšem popsán v následující kapitole.



Obrázek 5.3: Class diagram – manager (zdroj: autor)

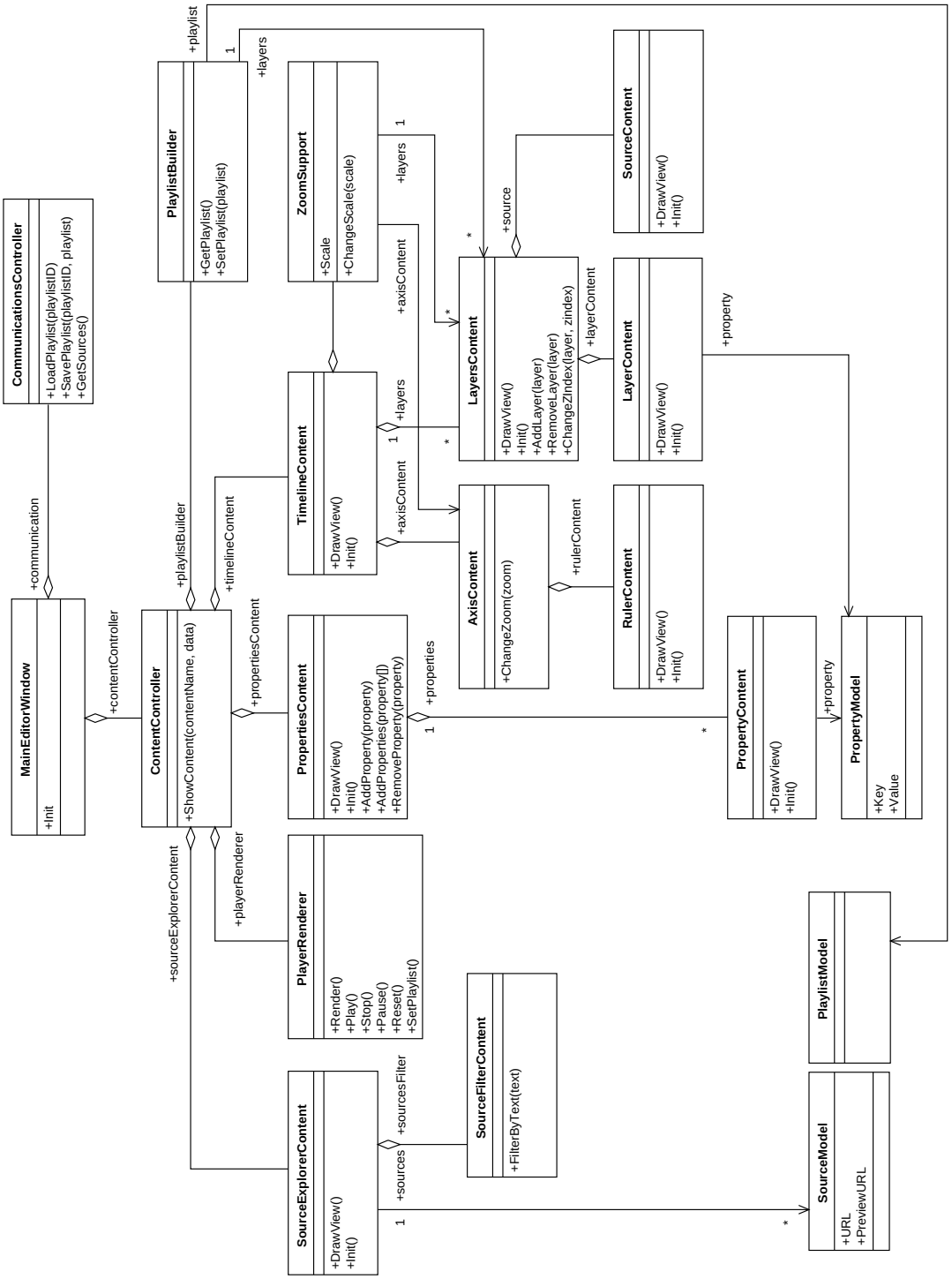
Editor scénáře

Aplikace editoru scénáře vychází ze třídy *MainEditorWindow*. Jedná se o vstupní spustitelnou třídu, která se postará o inicializaci a korektní spuštění všech potřebných částí. První potřebnou částí je třída *CommunicationController*, která se stará o komunikaci se serverem. Skrz tuto třídu se zajišťuje získání referencí na zdrojové soubory a přístup ke scénářům. Grafickou stranu zajišťuje třída *ContentController*, jenž zajistí korektní vykreslení všech potřebných grafických modulů aplikace. Třída *SourceExplorerContent* zpřístupňuje uživateli práci se soubory, zvláště pak jejich filtrování. *PlayerRenderer* je převzat ze samotného přehrávače a od diagramu 5.2 se liší pouze ve vynechání komunikačního modulu. Správu vlastností zdroje umístěného na časové ose zajistí třída *PropertiesContent*. Ta je určena pro vykreslení uživatelsky upravitelných vlastností a výběr správného uživatelského vstupu pro danou vlastnost – pro vlastnost barvy nabídne *color picker*, pro výčet možností nabídne *drop down list* apod.

Samotná časová osa je reprezentována třídou *TimelineContent*. Obsahuje časovou osu spolu s pravítkem. Rozsah celé osy je dynamicky počítán dle nastaveného přiblížení. Pokud se mění přiblížení, mění se i jemnost vykreslovaných časových úseků pravítka. O počítání správného přiblížení se stará třída *ZoomSupport*. Třída *LayersContent* reprezentuje zásobník jednotlivých vrstev. Jednotlivé vrstvy lze mezi sebou řadit do tzv. z-indexu[1]. Řazení z pohledu GUI probíhá pomocí systému *drag&drop*. Více o ovládání editoru scénáře v kapitole 5.5.1.

Neméně podstatnou třídou je *PlaylistBuilder*. Její potřeba nastává ve chvíli, kdy má dojít k přehrátí náhledu či uložení právě vytvořeného scénáře grafickým rozhraním. Zmíněná třída zde funguje jako převodní interface ze scénáře reprezentovaného pomocí grafických prvků na scénář popsaný v kapitole 4.4. Vytvoření celého scénáře je prováděno během jedné procedury a nikoliv přírůstkovými metodami.

[1] Z-index značí číselné označení hloubky na ose z – pomocí z-indexu se určuje překrývání prvků



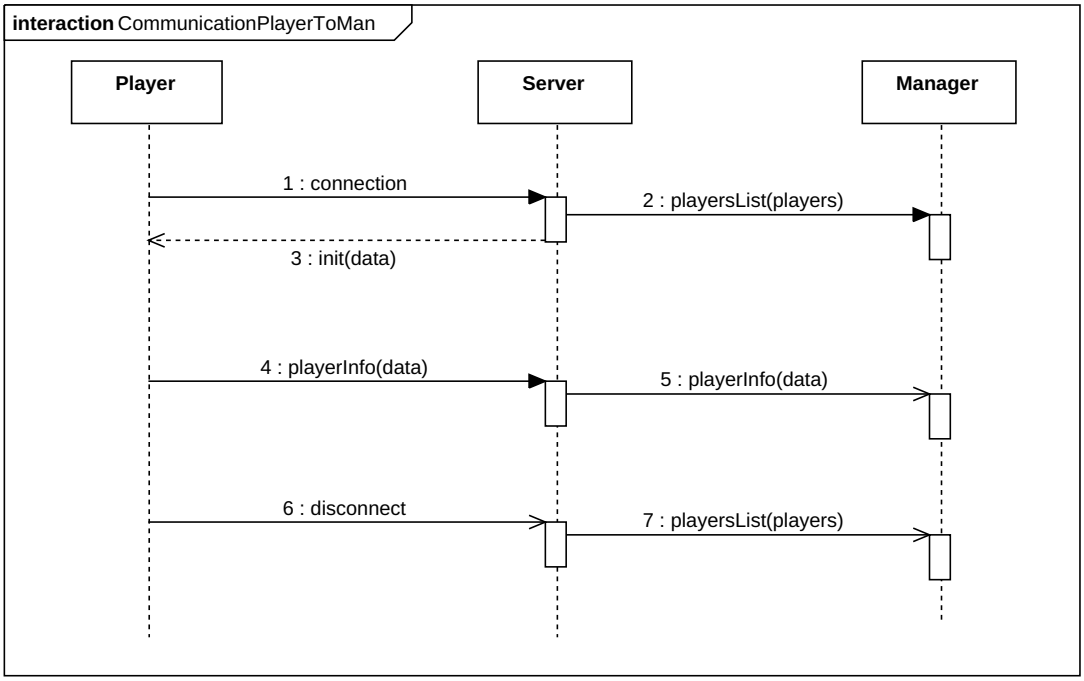
Obrázek 5.4: Class diagram – editor (zdroj: autor)

5.3 Síťová komunikace

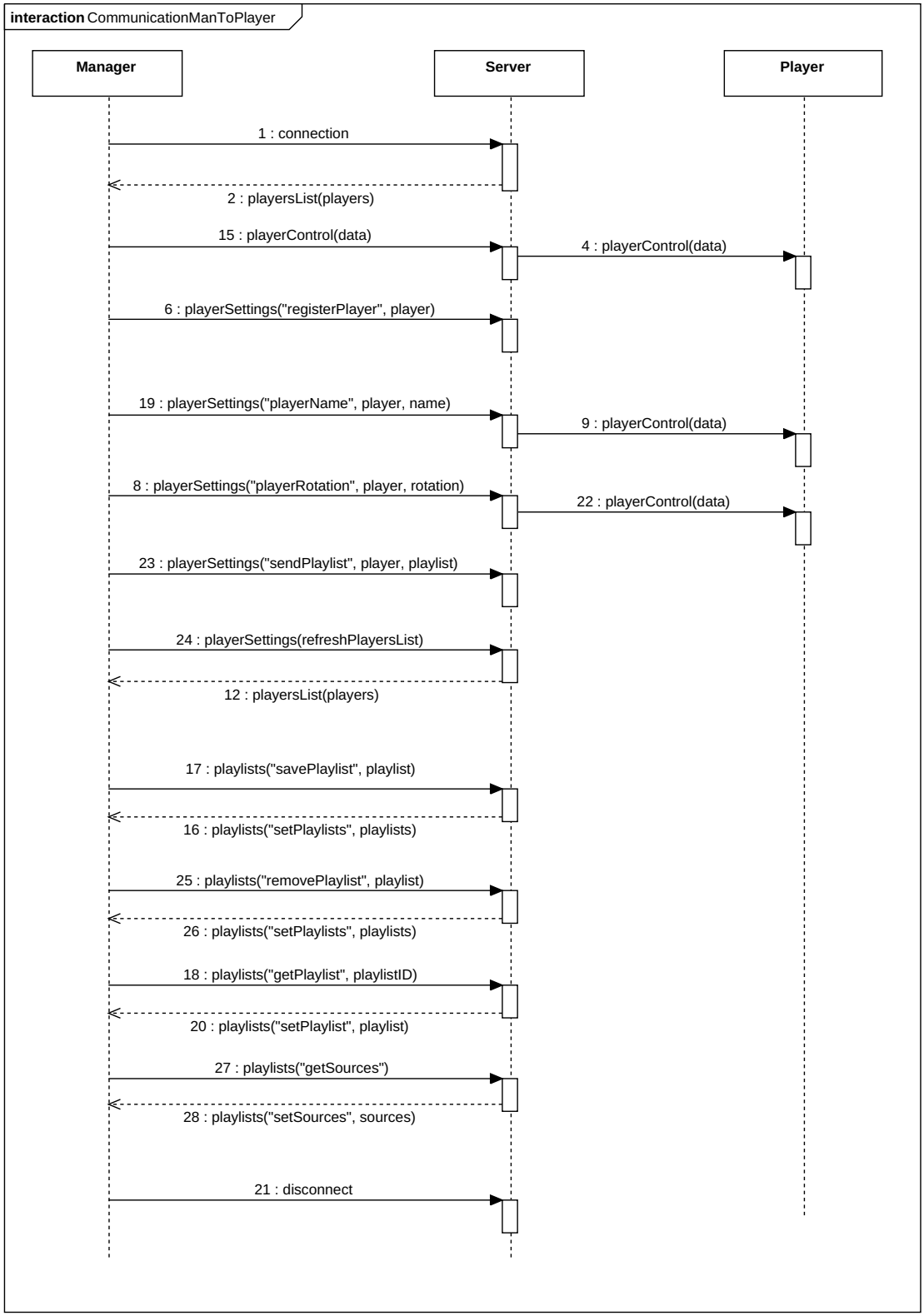
Síťová komunikace popsaná v této kapitole je realizací analytického návrhu a vychází ze seznamu požadavků. Samotná implementace je tvořena za pomoci knihovny socket.io (viz. kapitola 5.1.6). Spojením návrhu s uvedenou knihovnou vznikl pro komunikaci v reálném čase interface popsaný pomocí sekvenčních diagramů.

Sekvenční diagram 5.5 vyjádřuje síťovou komunikaci směřující od přehrávače směrem ke správci. Návrh zamezuje přehrávači kontaktovat správce napřímo, tzn. s vynecháním prostředního prvku – serveru. Síťová komunikace musí být vždy směrována skrze centrální server. Jak je z diagramu patrné, přehrávač má k dispozici omezené komunikační schopnosti. Prakticky se omezuje pouze na ohlášení připojení a hlášení aktuální činnosti, kterou zrovna vykonává.

Druhý sekvenční diagram 5.6 popisuje směr síťové komunikace od správce k přehrávačům. Z uvedeného diagramu lze vyvodit, že obsahuje události pro navázání spojení, správu přehrávačů a správu scénářů. Správce komunikuje opět pouze se serverem. V určitých případech pak server na základě dotazu od správce předává informace přehrávači. Rovněž, setjně jako v komunikaci od přehrávače, ani zde není povoleno decentralizování sítě. Je zcela zamezeno kontaktovat z klientské strany ostatní klienty – s přemostěním serveru.



Obrázek 5.5: Sekvenční diagram – směr od přehrávače ke správci (zdroj: autor)



Obrázek 5.6: Sekvenční diagram – směr od správce k přehrávači (zdroj: autor)

5.4 Server

Na základě požadavků server plní účel synchronizačního prvku a datového skladu. Synchronizace je hlavně ve smyslu předávání dat mezi aplikacemi a tím udržování aktuálních informací v rámci systému.

Datovým skladem je myšleno uchování zdrojů potřebných pro správnou funkci scénáře. Také jsou zde ukládány informace o přehrávačích. Lze tak listovat v seznamu přehrávačů i ve chvíli, kdy přehrávače nejsou připojeny pomocí síťové komunikace.

5.4.1 Identifikace klientských aplikací

Každá klientská aplikace (přehrávač a správce) má své jedinečné kódové značení, podle kterého probíhá identifikace zařízení. Označení je potřebné pro korektní komunikaci – zajištění, že data budou odeslána správnému cíli. Značení vzniká následujícím postupem: při otevření spojení se na serveru vytvoří nová relace, tzv. *session*. V ní jsou popsány informace o síťovém spojení a připojeném zařízení. Součástí session je i jedinečný identifikátor, který je v případě počáteční neexistence automaticky přidělen serverem. Klientská část si toto označení uloží do tzv. *cookie*[2] a odesílá ho s každým dalším dotazem na server. Tímto způsobem lze učinit jinak bezestavovou komunikaci řízenou protokolem *http* komunikací stavovou.

5.4.2 Ukládání dat na serveru

Ukládání dat na serveru je řešeno pomocí datových souborů. Bylo tak voleno z následujících důvodů: pro systém s několika přehrávači generuje poměrně malá data pro zapsání na disk. Nejčastěji se jedná o uchování logů. Pro koordinaci dotazů byla navržena třída, která řeší synchronizaci dat v paměti RAM s daty na pevném disku. Absencí databázového serveru se zjednodušuje instalace aplikace a snižují se tak potřebné závislosti při instalaci.

5.5 Správce

Ačkoliv je správce dostupný prostředictvím serveru jakožto webová stránka, jedná se o aplikaci běžící na klientské straně. Je napsána jako tzv. *tlustý klient* – aplikace obsahuje veškeré funkční omezení a pro samotný běh nepotřebuje dalšího partnera, od kterého by bylo třeba právě tyto funkční omezení čerpat. Aplikaci lze zkompilovat některým z nástrojů, které jí

[2] Cookie je značení lokální klientské paměti.

umožní fungovat jako nativní aplikace daného operačního systému. Jedním z těchto nástrojů je i tzv. *Apache Cordova*. Tento nástroj ovšem není součástí této práce. V případě zájmu si čtenář může opatřit více informaci na adrese [19].

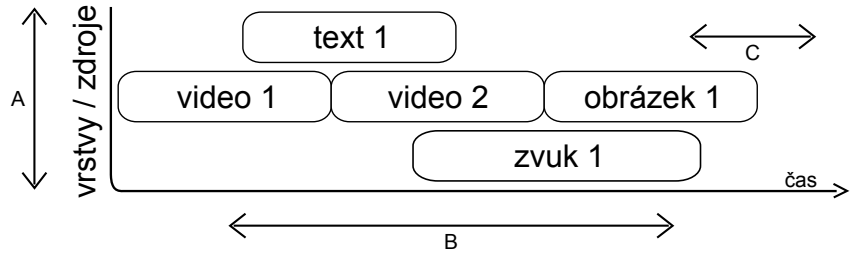
Z pohledu kódu je správce rozdělen do modulů, přičemž nejvyšším modulem je prvek, který spravuje jednotlivé uživatelské obrazovky. Určuje, která obrazovka bude zobrazena, a při přepnutí obrazovky zajišťuje veškeré operace s tím spojené. Pro možnost pohodlného přepínání mezi jednotlivými obrazovkami obsahuje hlavní menu, pomocí něhož je uživatel navigován. Každá obrazovka zajišťuje jednu ucelenou oblast a je napsána jako samostatný modul s implementovaným určitým rozhraním.

5.5.1 Editor scénáře

Z pohledu grafického rozhraní je kladen důraz na ovládání myší. Pro ovládání scénáře v čase se velmi hodí systém *drag & drop*, neboli *chyť a upusť*. Tento systém funguje na principu uchopení prvku kurzorem myši a následným pohybem myši se mění spolu s pozicí kurzoru i pozice uchopeného prvku. Zde využité možné pohyby jsou znázorněny na obrázku 5.7. Pohyby jsou naznačeny šipkami s označením pomocí velkého písmena (A, B, C).

Pohyb po ose *A* přináší možnost změny hloubky vykreslování zdrojových souborů. Čím hlouběji se zdroj nachází, tím je větší pravděpodobnost, že ho na obrazovce překryje zdroj jiný. Tzn. nejvýše umístěný zdroj bude vždy navrchu – vykreslen jako poslední. Příkladová implementace pohybu *A* v grafickém rozhraní je znázorněna v ukázkovém kódu 5.3. Ukázkový kód využívá funkce z externí knihovny rozšiřující knihovnu jQuery o možnosti řazení prvků pomocí kurzoru myši. Jedná se o knihovnu *jQuery UI – sortable*. Více o knihovně na [27].

Pohyby ve směru osy *B* i *C* vyjadřují pohyb v čase, přičemž pohyb *B* mění počátek zobrazení zdroje – v jakém čase se začne přehrávat video či se zobrazí určitý obrázek apod. Tento pohyb nemění délku zobrazení daného média. Pro změnu délky přehrávání určitého média je určen pohyb *C*. Tímto pohybem lze ovlivnit délku přehrávání videa, audia či zobrazení obrázku. Příklad implementace tohoto pohybu je popsán ve zdrojovém kódu 5.2. Ukázka je postavena nad knihovnou *jQuery UI – draggable*, která rozšiřuje funkce jQuery. Více o knihovně na [27].



Obrázek 5.7: Ukázka pohybu zdrojů v editoru scénáře (zdroj: autor)

Listing 5.2: Skript pro pohyb zdroje po časové ose (jazyk JavaScript)

```
// na vsechny prvky tridy "source" aplikuje funkci "draggable"
$(".source").draggable({
// podpora scrolovani
    scroll: true,
// typ kurzoru
    cursor: "move",
// osa pohybu x
    axis: "x",
// tato funkce se provolava pri tahu prvku mysi
    drag: function () {
        var position = $(this).position();
    // nastaveni pozice X
        X(position.left);
    // nastaveni pozice Y
        Y(position.top);
    },
// omezujici kontejner, kde se muze prvek pohybovat
    containment: "parent",
// prvky, ktere jsou vyjmuty z chyceni mysi
    cancel: "div.resize"
});
```

Editor scénáře obsahuje i systém pro úpravu vlastností jednotlivých zdrojových souborů umístěných na časové ose. Pokud se pomocí myši klikne na určitý zdroj, dojde pro jeho zviditelnění k podbarvení a aktivuje se funcionalita zajišťující úpravu vlastností. Aktivace probíhá formou předání všech dostupných vlastností zdroje jiné části aplikace. Ta projde příchozí vlastnosti, vybere uživatelsky editovatelné a individuálním přístupem je vykreslí do seznamu. Při vykreslení do seznamu je třeba znát i sémantiku jednotlivých vlastností, protože úprava barvy či výběr z číselníku probíhá z uživatelského rozhraní odlišně. Odlišnosti v úpravách musí být systémem podporovány. Dále spolu s vlastnostmi jsou předány i validační funkce. Pokud uživatel upraví stávající hodnotu, je zkontrolována patřičnou funkcí, která rozhodne o korektnosti úpravy. Tímto se ošetří správný uživatelský vstup. Zajistí se, aby textové pole odmítlo text, pokud je požadována číselná hodnota.

Když se vytvoří určitý scénář, je třeba uživateli nabídnout i jeho náhled. Náhledové okno využívá funkcionality samotného přehrávače s vypnutou přímou komunikací se serverem. Přehrávač je popsán v následující kapitole 5.6. Nyní podstatnější část tvoří generátor scénáře. Ze zdrojů umístěných na časové ose v grafickém rozhraní je třeba vyexportovat data ve tvaru popsaném v kapitole 3.1.4. To se děje na principu projití jednotlivých vrstev editoru scénáře a transformování dat do patřičné podoby, přičemž zde musí být přepočítána pozice zdrojů v uživatelském rozhraní na konkrétní snímky v konečném videu. Přepočet je

závislý na aktuálním přiblížení editoru scénáře a zvolené snímkové frekvenci. Dále musí být dodrženo správné vrstvení zdrojů, aby docházelo ke správému překrývání na obrazovce.

Listing 5.3: Skript pro řazení vrstev ve scénáři (jazyk JavaScript)

```
// na vsechny prvky tridy "table.layers" aplikuje funkci "sortable"
$("table.layers").sortable({
// prvek, kterym lze pohybovat s celym prvkem
    handle: ".layer-drag",
// kdyz se presouva prvek, je nahrazen timto prvkem
    placeholder: "layer-placeholder",
// pohyb v ose Y
    axis: "y",
// typ kurzoru
    cursor: "move",
// funkce, ktera je provolana na pocatku pri chyceni prvku mysi
    start: function (e, ui) {
        this.dataLastStartIndex = ui.item.index();
        this.dataLastSourceID = Object.keys(base.layers)[ui.item.
           index()];
    },
// funkce, ktera je provolana pri upusteni prvku
    stop: function (e, ui) {
        var newIndex = ui.item.index();
        var newSourceID = Object.keys(base.layers)[ui.item.index()];
        if (newSourceID != this.dataLastSourceID) {
            base.layers[this.dataLastSourceID].zindex(newIndex);
            base.layers[newSourceID].zindex(this.dataLastStartIndex)
               ;
            var layer = base.layers[sourceID];
            base.options.onLayerIndexChanged(
                    layer, this.dataLastStartIndex, newIndex);
        }
    }
// zakazani moznosti vyberu textu na strance
}).disableSelection();
```

5.6 Přehrávač

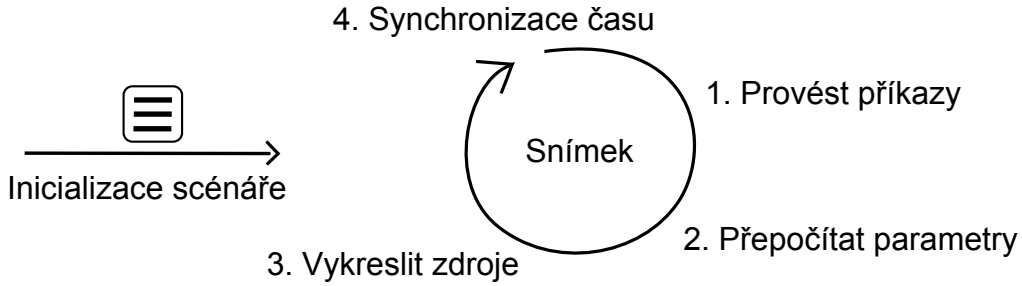
Podobně jako správce v předchozí kapitole 5.5, je i aplikace přehrávače napsána podobným způsobem. Lze ji zkompilovat některým z nástrojů, které ji umožní fungovat jako nativní aplikace daného operačního systému.

5.6.1 Vykreslovací jádro přehrávače

Vykreslovací jádro přehrávače přímo konzumuje scénář (popsaný v kapitole 4.7). Algoritmus znázorněný na obrázku 5.8 funguje následovně:

Při inicializaci scénáře se načtou a připraví veškeré zdroje.

Vykreslovací část poté funguje v nekonečné smyčce, kde jedna smyčka značí vykreslení jednoho snímku. V každém cyklu dojde ke zpracování veškerých příkazů ze seznamu událostí určených danému snímku (např. spustit – zaktivnit zdroj, nebo naopak deaktivovat zdroj). Dále pro každý aktivní zdroj (zdroj figurující v aktuálním snímku) se přepočítají parametry dle definovaných událostí pro daný snímek. Např. se změní průhlednost mezi snímkem předchozím a snímkem novým o určitý koeficient. Pokud jsou všechny aktivní zdroje přepočítány, dojde k vykreslení všech aktivních zdrojů ve výsledný snímek.



Obrázek 5.8: Práce vykreslovacího jádra (zdroj: autor)

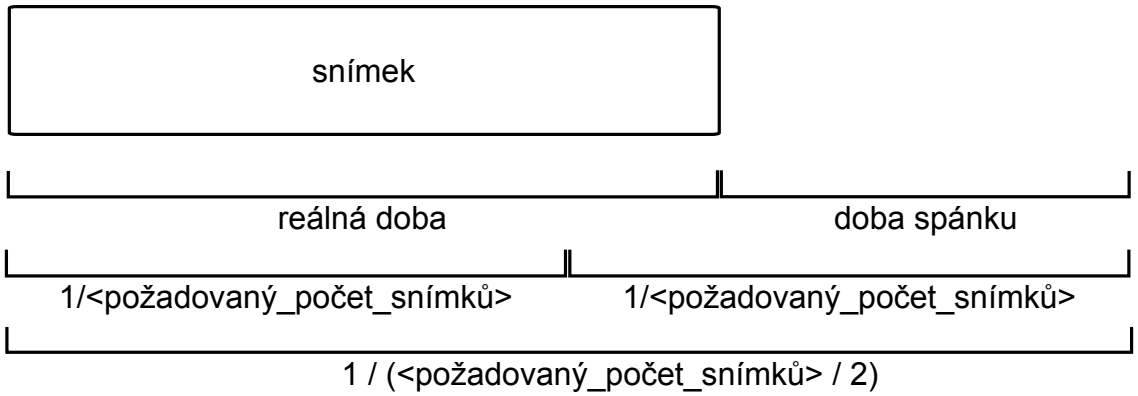
Posledním krokem je zajištění snímkové synchronizace, aby vykreslování odpovídalo scénářem nastavené snímkovací frekvenci. Výpočtem: 1/<požadovaný počet snímků> se zjistí maximální čas, který je přípustný pro vykreslení jednoho snímku. Pokud je změřený čas pro vykreslení snímku nižší, dojde k uspání renderujícího jádra na zbytek času (obr. 5.9). Pokud potřebný čas je vyšší než přípustný, musí dojít ke snížení snímkové frekvence. Kvůli ušetření času na výpočtu mezisnímků se postupuje po násobcích 2 (obr. 5.10).

Obrázek 5.9: Znázornění synchronizace času při vykreslování snímku (zdroj: autor)

Protože se jedná o multiplatformní aplikaci, je využita technologie HTML5. Ukázka zdrojového kódu 5.5 znázorňuje realizaci smyčky pro vykreslování jednotlivých snímků. Ukázka 5.4 řeší mezisnímkovou synchronizaci dle kapitoly 5.6.1. Popis algoritmu je řešen přímo v ukázce kódu.

K samotnému vykreslování je použit tzv. modul canvas, jenž je součastí specifikace HTML5. Tento modul sám o sobě poskytuje možnosti práce s 2D a 3D grafikou. Jeho hlavní předností je, že k samotnému vykreslování využívá hardwarovou akceleraci.

HTML5 má v sobě zabudovanou podporu dekódování videa i audia s využitím online streamingu. V případě videa jsou pak navenek poskytovány jednotlivé snímky, na které lze aplikovat grafické filtry či korektury barev. Následné přesměrování na výše zmíněný canvas s upřesňujícími informacemi se postará o vykreslení snímku na obrazovku.



Obrázek 5.10: Znázornění synchronizace času při vykreslování snímku s přetečením (zdroj: autor)

Listing 5.4: Mezi-snímková synchronizace (jazyk JavaScript)

```
function syncTime() {
  var diff = base.endFrameTime - base.startFrameTime;
  var framerate = playlist.settings.framerate | base.options.
     framerate;
  var timeout = 1000 / framerate;
  // volani funkce pro vykresleni snimku se synchronizovanym casem
  if (timeout > diff) {
    setTimeout(draw, timeout - diff);
  } else {
    setTimeout(draw, timeout * (Math.floor(diff / timeout) + 1));
  }
}
```

Listing 5.5: Smyčka pro kreslení snímků (jazyk JavaScript)

```
// vykreslovaci funkce
function draw() {
  // kontrola behu
  if (!base.isRunning) {
      setTimeout(draw, 100);
      return false;
  }
  base.startFrameTime = new Date();
  try {
    // vykresleni snimku
    drawFrame();
  } catch (ex) {
      logger.error("Draw␣frame␣error␣[" + base.frame + "].", ex);
  }
  // testovani konce scenare
  if (base.frame == playlist.timeline.frameTo) {
      base.reset();
  }
  base.endFrameTime = new Date();
  // synchronizace snimkovani
  syncTime();
}
```

5.6.2 Synchronizace správného poměru stran

Protože scénář využívá pro definici rozměrů vykreslovaného okna normovanou jednotku <0;1> (viz. kap. 3.1.4), je nutné tuto jednotku přepočítat na rozlišení vykreslovaného okna přehrávače. Tato činnost se děje na koncovém zařízení v rámci vykreslovacího jádra. Algoritmus přepočtu je k dispozici v ukázce zdrojového kódu 5.6.

Listing 5.6: Výpočet správného poměru stran (jazyk JavaScript)

```
// prepocet z normovanych jednotek <0;1> na realnou velikost okna
function fixAspectRatio (width, height, playlist) {
  // pomer stran obrazovky
  var r1 = width / height;

  var aspectX = playlist.settings.aspectX;
  var aspectY = playlist.settings.aspectY;
  // pomer stran playlistu
  var r2 = aspectX / aspectY;

  var obj = {
      dx: 0,
      dy: 0,
      width: width,
      height: height
  }
  // prizpusobeni sirky, ci vysky
  if (r1 < r2) {
      obj.width = width;
      obj.height = (1/r2) * width;
  } else {
      obj.height = height;
      obj.width = r2 * height;
  }
  // centrovani obrazu
  obj.dx = Math.abs(width - obj.width) / 2;
  obj.dy = Math.abs(height - obj.height) / 2;
  return obj;
}
```

6 Nasazení a testování

Aplikace server potřebuje pro správnou funkčnost nainstalovanou technologii Node.js. Popis instalace a spuštění jednotlivých aplikací je popsán v sekcích 6.2.1 pro Ubuntu a 6.2.2 pro Microsoft Windows.

6.1 Požadavky na zařízení

Podmínkou pro zařízení, kde má být spuštěna aplikace přehrávač, je nutná podpora grafického rozhraní operačním systémem a podpora hardwarové akcelerace grafických prvků. Součástí operačního systému musí být instalována i poslední verze webového prohlížeče *Mozzilla Firefox* [20]. Ostatní webové prohlížeče nemusí splňovat plnou podporu.

Na základě potřebného rozsahu systému lze volit dva přístupy v přístupu k systému:

1. Jediné zařízení
2. Více zařízení

6.1.1 Jediné zařízení

Je-li k dispozici pouze jediné zařízení, může fungovat jako server i přehrávač zároveň. Podmínkou pro takovéto zařízení je nutná podpora grafického rozhraní operačním systémem a podpora hardwarové akcelerace grafických prvků. Na toto zařízení se vysadí aplikace server – postup vysazení a spuštění je v kapitole 6.2. V internetovém pohlížeči poté stačí načíst webovou adresu odpovídající tvaru uvedenému v tabulce 6.1 a přehrávač je připraven k provozu. Ovládání a nastavení přehrávače je pak nutné provést z jiného zařízení opět pomocí webového prohlížeče za pomoci webové adresy odpovídající příslušnému záznamu v tabulce 6.1.

6.1.2 Více zařízení

Výhodnějším řešením je použití více zařízení, tedy samostatných zařízení pro jednotlivé přehrávače. Postup vysazení je podobný jako v kapitole 6.1.1. Liší se pouze o to, že aplikace

přehrávač má své vlastní zařízení a nesdílí tak zařízení s aplikací server. Aplikace přehrávače se distribuuje pomocí webové adresy *http://<ip-adresa-serveru>:<port>/player*.

Tabulka 6.1: Porovnání aplikací přístupných skrze centrální server

Aplikace	Obecná adresa	Výchozí adresa
Player	http://<ip-adresa-serveru>:<port>/player	http://localhost:3020/player
Manager	http://<ip-adresa-serveru>:<port>/manager	http://localhost:3020/manager

6.2 Instalace aplikace

6.2.1 Instalace na operační systém Ubuntu

Veškerá instalace a ovládání probíhají pomocí terminálu. Pro instalaci technologie Node.js je třeba spustit příkazy 6.1. Poté proběhne spuštění aplikace pomocí příkazů 6.2.

Listing 6.1: Instalace Node.js na operační systém Ubuntu

```
# stazeni technologie Node.js
curl -sL https://deb.nodesource.com/setup_5.x | sudo -E bash -
sudo apt-get install -y nodejs
sudo apt-get install -y build-essential
```

Listing 6.2: Spuštění serveru na operačním systému Ubuntu

```
# spusteni serveru
cd /<cesta-do-rootu-aplikace>
# spusteni s vypisem do konzole
node src/app.js

# pro korektni spusteni na pozadi je treba aplikace Forever
# instalace aplikace Forever
npm install forever -g
# spusteni na pozadi (produkcni beh)
sh startServer.sh
```

6.2.2 Instalace na operační systém Microsoft Windows

Technologie Node.js je ke stažení na adrese [17]. Doporučenou verzí je 64-bit v5.x.x. Instalace je řešena klasickým grafickým instalátorem. Po úspěšném nainstalováním je třeba spustit CommandLine (popř. PowerShell), ve kterém se spustí příkazy 6.3.

Listing 6.3: Spuštění Node.js na operačním systému MS Windows

```
:: spusteni aplikace
dir <cesta-do-rootu-aplikace>
:: spusteni s vypisem do konzole
node src/app.js

:: pro korektni spusteni na pozadi je treba aplikace Forever
:: instalace aplikace Forever
npm install forever -g
:: spusteni na pozadi (produkcni beh)
forever --append -l "./forever.log" -o "./out.log" -e "./err.log" --
   minUptime 1000 --spinSleepTime 1000 start --uid "server" "src/app
   .js"
```

6.3 Testovací scénáře

Testovací scénáře poslouží jako ukazatele, zda jsou určité funkce implementovány v souladu s případy užití. Pomocí testovacích scénářů je snaha nasimulovat systému různé podmínky podobné produkčnímu prostředí. Veškeré chování systému je nutné při testování scénáře sledovat, protože v případě neúspěšné realizace testovacího scénáře to pomůže v následném hledání chyb.

V průběhu vývoje byly testovány veškeré implementované případy užití. Výčet některých testovacích scénářů je k nahlédnutí v tabulkách: 6.2, 6.3, 6.4 a 6.5.

Tabulka 6.2: Testovací scénáře pro UC_04 Vytvoření scénáře

ID	Scénář	Očekávaný výsledek
TC_04_01	Vytvořit scénář	Systém zobrazí editor scénáře
TC_04_02	Vytvořit scénář bez navázání spojení se serverem	Systém informuje o ztrátě spojení se serverem.
TC_04_03	Vytvořit scénář - pojmenování scénáře s více než 300 znaky	Systém uloží scénář a zobrazí seznam všech uložených scénářů.

Tabulka 6.3: Testovací scénáře pro UC_06 Upravit nastavení přehrávače

ID	Scénář	Očekávaný výsledek
TC_06_01	Změnit název přehrávače	Jméno uloží na serveru a příslušném přehrávači.
TC_06_02	Změnit úhel natočení přehravače	Úhel natočení uloží na serveru a příslušném přehrávači. Přerávač bude renderovat pootočený obraz.
TC_06_03	Změnit název u nepřipojeného přehrávače	Systém odmítne změnu provést
TC_06_04	Změnit název u nepřipojeného přehrávače	Systém odmítne změnu provést

Tabulka 6.4: Testovací scénáře pro UC_14 Přidat zdroj

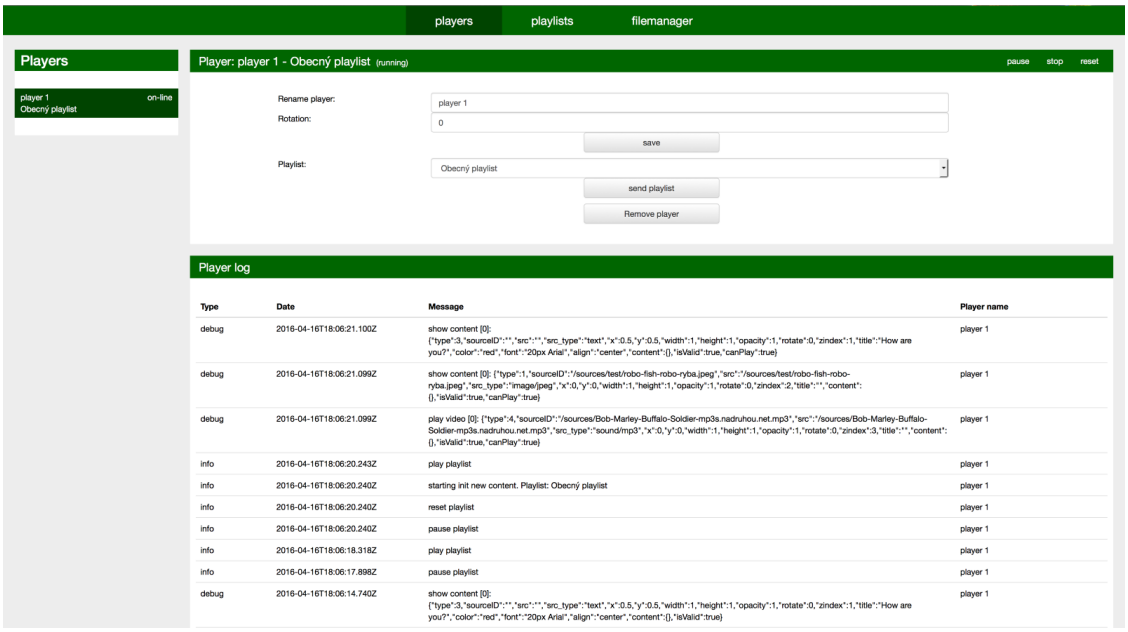
ID	Scénář	Očekávaný výsledek
TC_14_01	Přidat zdroj skrze nativní výběr souborů prohlížeče	Zobrazí dialogové okno se souborovým systémem a vybraný soubor uloží na server
TC_14_02	Přidat zdroj systémem drag&drop ze správce souborů	Dojde ihned k odeslání zdroje na server, který zdroj uloží
TC_14_03	Přidat zdroj jakýmkoliv způsobem, když není k dispozici server	Aplikace odmítne provést činnost

Tabulka 6.5: Testovací scénáře pro UC_16 Upravit vlastnosti vrstvy

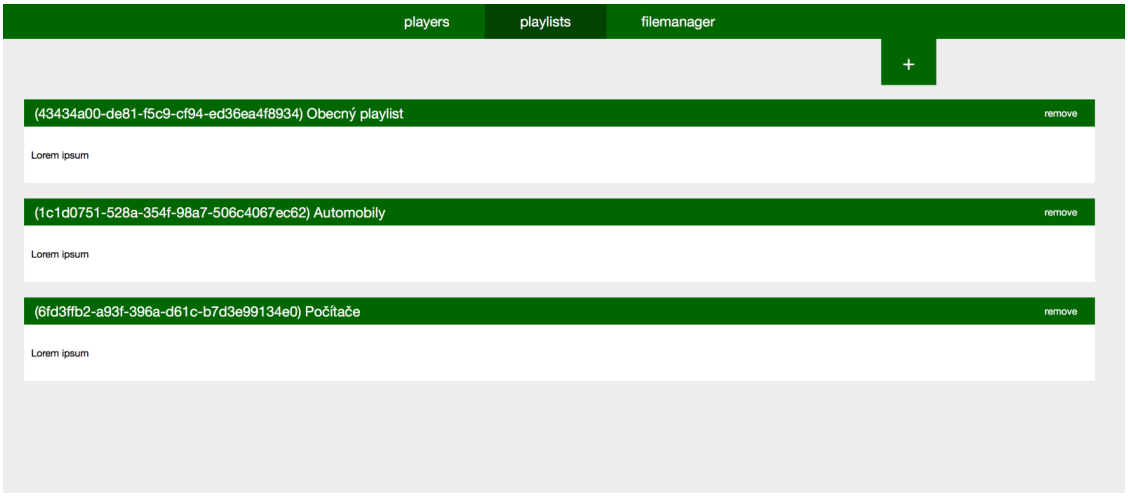
ID	Scénář	Očekávaný výsledek
TC_16_01	Upravit pozici na obrazovce v obou osách	Systém přijme pouze rozsah <0;1>.
TC_16_02	Upravit počátek přehrávání vrstvy	Systém omezuje vstup na interval <0;délka_scénáře>.
TC_16_03	Upravit délku zobrazení vrstvy	Systém omezuje vstup na interval <start_vrstvy;délka_scénáře-start_vrstvy>.
TC_16_04	Změna barvy	Systém omezuje vstup na hexadecimální tvar. K dispozici je grafický pomocník pro výběr barvy.
TC_16_05	Změna textu v textovém poli	Systém dovoluje vložit text s minimálně jedním libovolným znakem.
TC_16_06	Změna hlasitosti u vrstev podporujících audio	Systém přijme pouze rozsah <0;1>.

6.4 Náhled aplikací

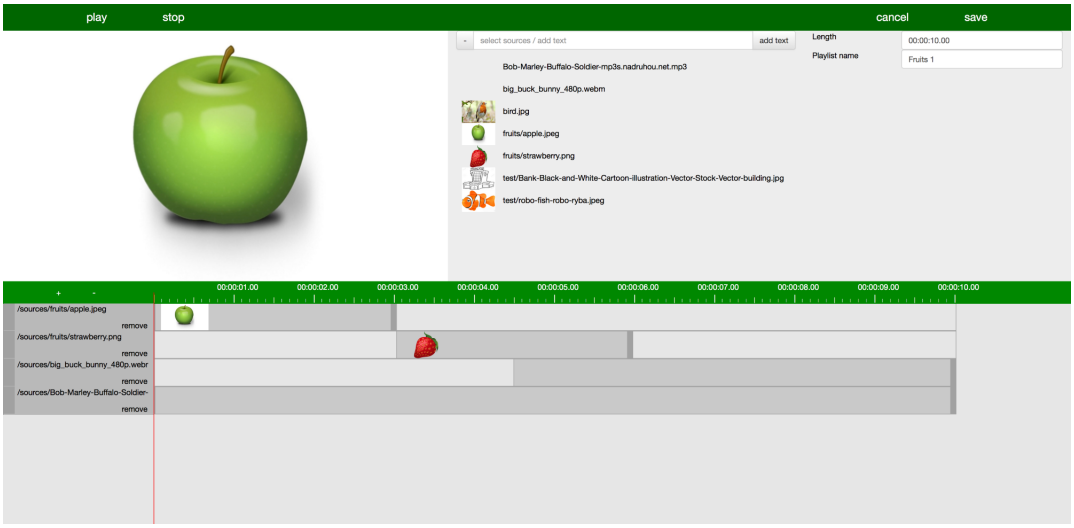
Na obrázku 6.1 je grafický interface správy přehrávačů. Obrázek 6.2 vyobrazuje grafický interface správy scénářů. Editor časového scénáře je na obrázku 6.3. Aby mohl editor scénáře fungovat, musí být nabídnuta možnost spravovat zdrojové soubory. Pro správu souborů na serveru je určen správce souborů na obrázku 6.4.



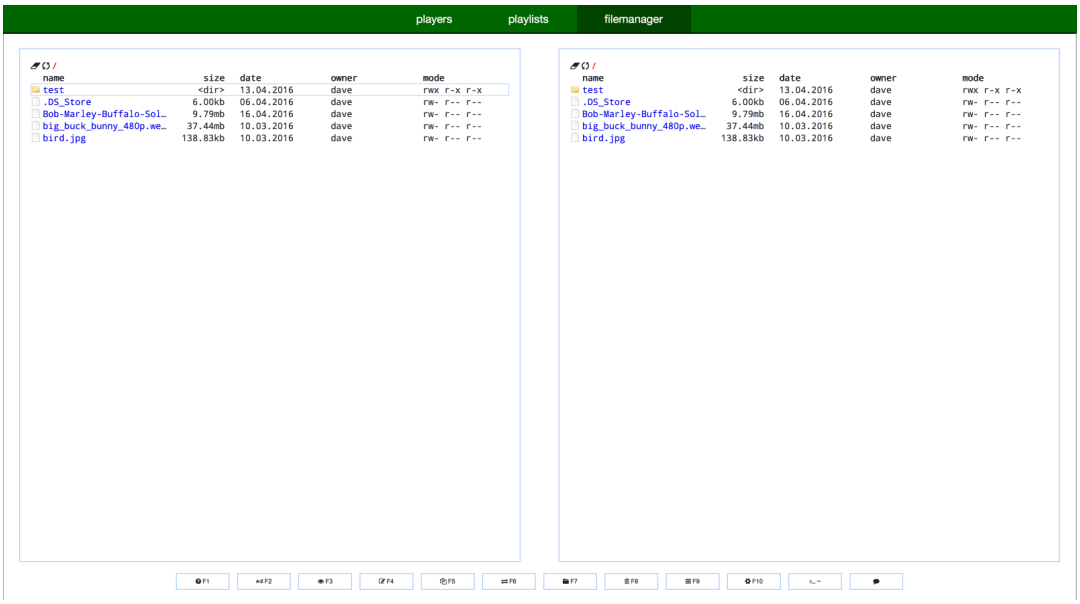
Obrázek 6.1: Náhled GUI – správa přehrávačů (zdroj: autor)



Obrázek 6.2: Náhled GUI – správa scénářů (zdroj: autor)



Obrázek 6.3: Náhled GUI – editor scénáře (zdroj: autor)



Obrázek 6.4: Náhled GUI – správce souborů (zdroj: autor)

7 Závěr

V této diplomové práci jsem se v první etapě zaměřil na teoretické zpracování problematiky. Popsal jsem systém pohledem z různých směrů a definoval způsoby užívaní. V druhé fázi jsem prostudoval specifikace již existujících systémů, které určitou formou prezentuji v této práci. Na základě specifikací cizích systémů jsem spolu s vlastními zkušenostmi s podobnými implementacemi provedl analýzu, ze které vznikl návrh systému nového. V návrhu byly definovány požadavky na systém, případy užití a scénáře případů užití. Byla navržena komunikace popsaná pomocí sekvenčních diagramů a samotný objektový návrh byl vypracován formou analytických diagramů. Z návrhu také vyplynul požadavek na rozdělení systému na tři samostatné aplikace.

Další má činnost byla zaměřena na samotnou implementaci mého návrhu. Serverovou část navrhovaného systému jsem se rozhodl implementovat v technologii Node.js. Klientské části poté jako tzv. *tlusté klienty* napsané pod specifikací HTML5 s rozšiřujícími JavaScriptovými knihovnami *Bootstrap* a *jQuery*. Pro vzájemnou komunikaci jsem využil knihovny *Socket.IO*, která zajišťuje komunikaci v reálném čase.

Implementací se podařilo dosáhnout funkčního propojení. Jednotlivé části mezi sebou interagují a reagují na jednotlivé podněty přicházející skrz komunikační síť. Systém nabízí uživateli rozhraní ke správě připojených zařízení. Implementovaný systém umožňuje sestavit pomocí uživatelských příkazů vstupujících přes GUI aplikace scénář. Následně nabízí uživateli možnost distribuce vytvořeného scénáře na konkrétní zařízení. Zvolené zařízení převezme scénář a je schopno ho reprodukovat – kontinuálně vykreslovat jednotlivé snímky v reálném čase. Scénář smí obsahovat statické obrazové materiály, video formáty, audio formáty a texty. Obsah se pro tvorbu scénáře musí umístit na server. K tomuto účelu slouží implementovaný správce souborů. Veškerý zmíněný přípustný obsah lze umístit poměrově na přesné místo na zobrazovacím zařízení a lze jej namapovat na časovou osu – vymezit tím časové okno, kdy bude daný obsah viditelný. Obsah je vykreslován ve vrstvách, tudíž lze ovlivnit způsob překrývání. Obsahuje-li obsah zvukovou stopu, lze jí nastavit hlasitost.

Prostor pro vylepšení je ve formě implementace podpory uživatelských účtů a s tím spojených vymezení uživatelských rolí. Dále, ačkoliv samotný scénář i jádro vykreslující obsah mají implementován způsob práce s animacemi, editor scénáře má v tuto chvíli funkci značně omezenou. Nyní umí nastavit jednu ukázkovou animaci pro zobrazení obrázku, videa a textu. Stejně tak podporuje pouze jednu ukázkovou animaci pro schování obrázku, videa a textu. Uživatel je tak sice schopen vytvářet jednoduché přechody, nicméně pro sku-

tečný chod to vyžaduje mnohem více typů animací. V neposlední řade je významným nedostatkem nedostačující zabezpečení síťového spojení, které ovšem není předmětem této práce.

Dále je třeba dokončit předrenderování scénáře. Tato funkcionalita byla částečně implementována a výsledek byl vykreslen do sady obrázků. Nepodařilo se ovšem korektně spojit zvukovou stopu s obrázky při vytváření konečného video souboru. Z důvodu časové náročnosti odladění této funcionality je aktuálně k dispozici pouze render videa v reálném čase.

Systém byl testován na základě vytvořených testovacích scénářů. Výstupy testů ukazují, že je systém běhuschopný ve webovém prohlížeči Mozilla Firefox.

Seznam použité literatury

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

[2] PETZOLD, Charles.: *Mistrovství ve Windows Presentation Foundation.* Brno: Computer Press, 2008, 928 s. Mistrovství. ISBN 978-80-251-2141-2.

[3] Javascript: *Dokumentace javascriptu* [online]. 2016 [cit. 2016-04-01] `https://developer.mozilla.org/en-US/docs/Web/JavaScript`

[4] jQuery: *Javascriptová knihovna - dokumentace* [online]. 2016 [cit. 2016-04-01]. `http://jquery.com/`

[5] Digital signage today: *Glog s novinkami v oblasti digitálních přehrávačů* [online]. 2016 [cit. 2016-04-01]. `http://www.digitalsignagetoday.com/blogs/`

[6] Xibo - Digital Signage: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `http://xibo.org.uk/`

[7] Xibo - Digital Signage - GitHub repozitář: *Oficiální stránky zdrojových kódů produktu* [online]. 2016 [cit. 2016-04-01]. `https://github.com/xibosignage/xibo`

[8] Scala: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `http://scala.com/`

[9] WireSpring: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `https://www.wirespring.com/Solutions/digital_signage.html`

[10] Embed Signage: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `https://embedsignage.com/`

[11] Screenly: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `https://www.screenlyapp.com`

[12] signagelive: *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01]. `https://signagelive.com/`

[13] AGPLv3: *Oficiální stránky licence* [online]. 2016 [cit. 2016-04-01].
http://www.gnu.org/licenses/agpl-3.0.html

[14] Domain Master: *Webové stránky certifikované autority* [online]. 2016 [cit. 2016-04-01].
https://www.domainmaster.cz/napoveda/caste-dotazy/ssl-certifikaty

[15] Certifikační autorita PostSignum: *Oficiální stránky certifikační autority PostSignum* [online]. 2016 [cit. 2016-04-01].
http://www.postsignum.cz/kvalifikovane_certifikaty.html

[16] Certifikační autorita PostSignum: *Oficiální stránky certifikační autority PostSignum* [online]. 2016 [cit. 2016-04-01].
https://cs.wikipedia.org/wiki/Elektronicky_podpis

[17] Node.js: *Oficiální stránky Node.js* [online]. 2016 [cit. 2016-04-01].
https://nodejs.org/

[18] Stažení Node.js: *Oficiální stránky Node.js* [online]. 2016 [cit. 2016-04-01].
https://nodejs.org/en/download/

[19] Apache Cordova *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://cordova.apache.org/

[20] Mozilla Firefox *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://www.mozilla.org/en-US/firefox/products/

[21] Express - web API server pro Node.js *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
http://expressjs.com/

[22] Cloud CMD - správce souborů pro knihovnu express *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://www.npmjs.com/package/cloudcmd

[23] HTML5 *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://www.w3.org/TR/html5/

[24] CSS *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://www.w3.org/Style/CSS/

[25] Socket.IO *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
http://socket.io/

[26] npm - balíčkovací systém *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://www.npmjs.com/

[27] jQuery UI *Oficiální stránky produktu* [online]. 2016 [cit. 2016-04-01].
https://jqueryui.com

A Obsah přiloženého CD

soubor/adresář	obsah
diplomova-prace.pdf	originální text diplomové práce
diagramy-trid/	složka s diagramy tříd
diagramy-trid/diplomova-prace.mdj	soubor s UML návrhem struktury zpracovaný v program StarUML
prakticka-cast/	složka se zdrojovými kódy
programatorska-dokumentace/	složka s programátorskou dokumentací
uzivatelska-dokumentace/	složka s uživatelskou dokumentací
uzivatelska-dokumentace/diplomova-prace.mp4	soubor s prezentací uživatelské práce