České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra počítačové grafiky a interakce

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Ondřej Zeman**

Studijní program: Otevřená informatika
Obor: Softwarové inženýrství

Název tématu: **Tvorba uživatelského rozhraní pro systém biofeedbackové terapie**

Pokyny pro vypracování:

Cílem práce je návrh komplexního uživatelského rozhraní, které slouží k ovládání průběhu biofeedbackové terapie. Práce je součástí rozsáhlejšího projektu pro administraci a správu této terapie.
Student provede analýzu současných systémů pro správu terapií a provede rozbor slabých míst. Následně vytvoří návrh uživatelského rozhraní, které umožní efektivní řízení terapie a její adaptivní nastavení terapeutem. Rozhraní bude konzultováno s terapeuty a přizpůsobeno jejich požadavkům. Terapeutické rozhraní obsahuje systém pro vizualizaci aktuální terapie s možností nastavování parametrů pro konkrétního pacienta. Databázové rozhraní umožní správu databáze klientů s možností vizualizace dlouhodobých trendů. Uživatelské rozhraní klienta bude reflektovat principy jednoduchosti a intuitivnosti, přičemž bude testováno na konkrétních klientech během cvičných terapií.
Student implementuje obě rozhraní do stávajícího projektu, provede jeho otestování a výsledky popíše v diplomové práci.

Seznam odborné literatury:

Saeid Sanei, J.A. Chambers. EEG Signal Processing. Wiley-Blackwell (an imprint of John Wiley & Sons Ltd)

Xavier J. Caro, Earl F. Winter. EEG Biofeedback Treatment Improves Certain Attention and Somatic Symptoms in Fibromyalgia : A Pilot Study. Science + Business Media, LLC 2011. Published online : 9 June 2011.

Vedoucí: Mgr. Michal Vavrečka, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

L.S.

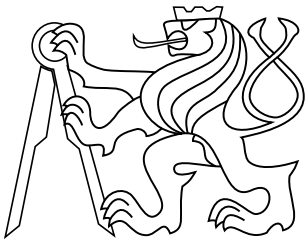
pr

V Praze dne 3. 3. 2014

diplomová práce

Tvorba uživatelského rozhraní pro systém biofeedbackové terapie

Ondřej Zeman



Květen 2015

Vedoucí práce: Mgr. Michal Vavrečka, Ph. D.

České vysoké učení technické v Praze
Fakulta elektrotechnická, Katedra kybernetiky

Poděkování

Mé poděkování patří především vedoucímu mojí práce Mgr. Michalovi Vavrečkovi, Ph. D. za jeho podporu a poskytnuté rady v průběhu tvorby mé práce. Poděkovat chci za jeho pomoc i Ing. Radimu Bělobrádkovi, za konzultace ohledně návrhu a provádění testování. Děkuji i svým přátelům za pochopení a podporu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Abstrakt

Tato práce se zabývá vytvořením komplexního grafického uživatelského rozhraní nad existujícím jádrem neurofeedbackové aplikace. Teoretická část práce zahrnuje východiska praktické části. Obsahuje informace o mozku, jeho projevech a jejich měření a dále také konkrétně popisuje neurofeedback, jeho definici, využití a přínos. Dále obsahuje popis modulární architektury aplikace s důrazem na intermodulární komunikační protokol. Zmiňuje i konkrétní technologie použité v aplikaci a jejich výhody. Kompletní analytická dokumentace obsahuje případy užití konkrétních modulů. Praktická část už se zabývá implementací jednotlivých modulů v aplikaci. K dispozici jsou přímo i ukázky uživatelského rozhraní. Práce uvádí způsob testování uživatelského rozhraní. Práce se dále zabývá budoucími možnostmi rozvoje aplikace jako celku a jejím celkovým přínos. Součástí práce je také CD s kompilovanými kódy.

Klíčová slova

biofeedback, grafické uživatelské rozhraní, GUI

Abstrakt

This thesis is focused on creation of complex graphical user interface running over existing core of neurofeedback application. The theoretical part represents the entry points for the practical part. It consists of information regarding brain, its functions and how they are measured. More specifically the neurofeedback is described – its definition, uses and benefits. Furthermore the description of modular architecture of application is mentioned. The specific technologies used while creating the application are listed together with the reasons for their use and their main advantages. The complete analytic documentation specifies the use cases of the modules. The practical part is focused on the implementation of the modules included in the application. There are also the examples of the user interface. The future possible development of the application is mentioned as well as the benefits it brings. In the attachment of the thesis there is a CD with compiled codes.

Keywords

biofeedback, graphical user interface, GUI

Obsah

Seznam obrázků

Zkratky

Sezam použitých zkratek

ADHD	Hyperkinetická porucha (Attention Deficit Hyperactivity Disorder)
CT	Počítačová tomografie (Computer Tomography)
DTO	Objekty pro přenos dat (Data Transfer Object)
GUI	Grafické uživatelské rozhraní (Graphical User Interface)
OMT	Techniky objektového modelování (Object-modeling Techniques)
OOSE	Objektově orientované softwarové inženýrství (Object-Oriented Software Engineering)
UML	Univerzální modelovací jazyk (Universal Modeling Language)

1 Úvod

V dnešní době pokročilých a rozvinutých technologií dochází k čím dál většímu pokroku propojením medicíny a techniky. Toto propojení nám umožňuje lépe poznat lidský mozek, jeho projevy a také pomáhá v léčení různých poruch. Moje práce spadá do tohoto průniku dvou světů – světa medicíny a světa moderních technologií. Biofeedback je moderní metodou, která využívá měření pochodů lidského těla. Specificky neurofeedback měří elektrické vlny mozku. Moje práce je součástí projektu, který vytváří aplikaci sloužící pro terapeuty a jejich klienty k zaznamenávání průběhu terapií a zpracovávání naměřených dat. Mým úkolem v rámci projektu je vytvořit uživatelské rozhraní této aplikace. Projekt má velký potenciál a aplikace díky svému propojení je jedinečnou a do budoucna velmi zajímavou cestou pro využití biofeedbacku. I v budoucnu bude umožňovat rozšiřování o další složitější moduly. Moje práce je rozčleněna na teoretickou a praktickou část, celkem se skládá ze sedmi kapitol.

V teoretické části se zabývám východisky svojí práce. První kapitola je zaměřena na věci související s neurofeedbackem – popisuje mozek, jeho projevy, a jak se tyto projevy měří. Dále se zabývá už konkrétně neurofeedbackem, jeho historií, použitím a přínosem. V další části je popsán návrh modulární architektury aplikace a intermodulárního komunikačního protokolu, který umožňuje jednotlivým modulům spolu efektivně komunikovat a sdílet data. Je zmíněn také stav původního jádra aplikace a je provedeno srovnání návrhu s podobnými existujícími aplikacemi. Ve třetí kapitole se zabývám technologiemi použitými při implementaci aplikace, jejich výhodami a důvody, proč byly použity. V následující kapitole najdeme analytickou dokumentaci, která obsahuje funkční popis aplikace, jednotlivých modelů a návaznost jejich případů užití.

Praktická část začíná implementací. Zde můžeme najít detaily implementace jednotlivých modulů. Zahrnuje konkrétní popis stavby a systému fungování jednotlivých modulů a jejich vzájemné komunikace a interakce. V šesté kapitole jsou uvedeny ukázky uživatelského rozhraní. Závěrem praktické části je testování a budoucí vývoj aplikace. Zahrnuje možné body, jakou cestou by se mohla tato aplikace dále vyvíjet.

Moje práce obsahuje především technické specifikace technické povahy a ve stručnosti shrnuje teoretické základy z oblasti medicíny nutné pro správné pochopení problematiky.

Cílem mé práce je vytvořit uživatelské rozhraní nad existujícím jádrem aplikace. Využívám informace teoretické dostupné na internetu a v bibliografických zdrojích v teoretické části, a v praktické hlavně svoje znalosti a zkušenosti v programování.

2 Neurofeedback

V rámci této kapitoly se budu zabývat hlavně teoretickými východisky práce z hlediska psychologického a medicínského. Tyto poznámky je nutné zmínit pro komplexní porozumění tématu a diplomové práce jako celku. Součástí této kapitoly je popis mozku jako takového, jeho částí a funkcí. Dále navazuje informace o mozkových vlnách a o tom jak se měří. Závěrečná část je věnována neurofeedbacku – jeho definici, použití a přínosu. Jedná se o východisko praktické části mojí diplomové práce.

2.1 Mozek

Tato část se zabývá základními poznatky o mozku, nervovém systému a mozkových vlnách.

Mozek je nejkomplexnějším a zároveň nejdůležitějším orgánem v těle. Kůra mozková má vliv na jedinečnost člověka – pocházejí z ní například myšlenky, jazyk, lidské vědomí nebo schopnost myslet a představovat si. Mozková kůra je to, co vidíme při pohledu na mozek. Můžeme jí rozdělit na 4 laloky – čelní, temenní, spánkový a týlní (Viz obrázek 1). Čelní lalok (frontal lobe) je část mozku zodpovědná za motoriku, rozum a vnímání. Změny této části mozku mohou vést ke změně v socializaci, pozornosti a vnímání rizika. Temenní lalok (parietal lobe) zpracovává smyslové vnímání těla. Jeho poškození vede k problémům s pamětí, koordinací očí. Spánkový lalok (temporal lobe) zahrnuje řečové a paměťové centrum, kdy jeho poškození vede k poruchám těchto funkcí. Týlní lalok (occipital lobe) zpracovává zrakové vjemy.[1]

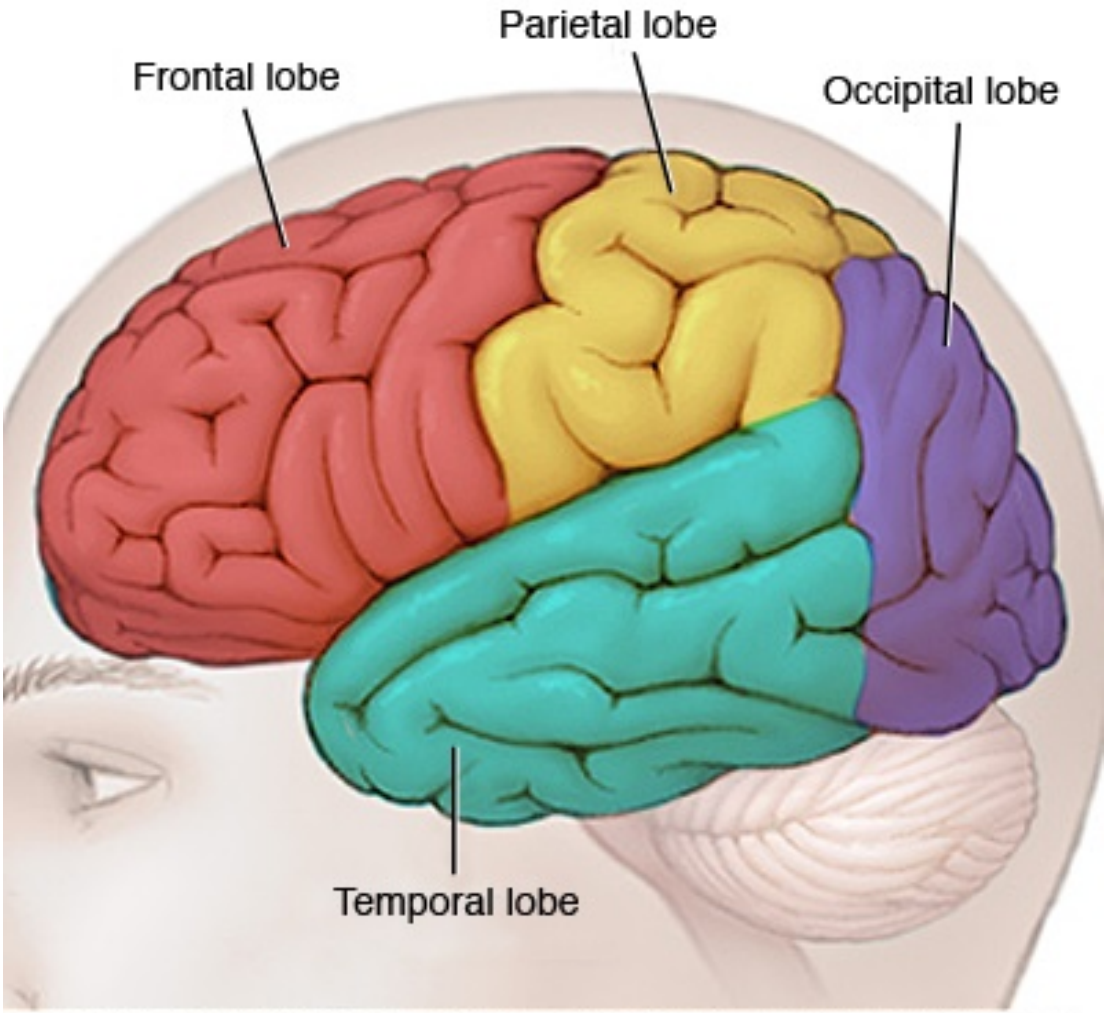
Mozek se dělí na dvě hemisféry – pravou a levou. Každá kontroluje opačnou část těla a jsou charakteristické rozdílnými kognitivními funkcemi. U většiny populace kontroluje levá hemisféra logiku a matematiku a pracuje s detaily, pravá hemisféra kontroluje intuici, emoce a pracuje s celkovým obrazem.[2]

Mozek se na mikroskopické úrovni skládá z neuronů a gliových buněk, které jsou základními stavebními kameny mozku. Základním úkolem neuronů je přenos vzruchu na další neurony. Tímto přenosem vznikají konkrétní funkce – třeba pohyb končetin. Neuron má obvykle jeden axon, tělo a vícero dendritů. Přes dendrity jsou vzruchy přijímány, přes tělo jdou na axon a odtud dále k dalším neuronům. Místu kontaktu dvou neuronů se říká synapse. Větší celky neuronů tvoří jádra. Gliových buněk je více než neuronů a to na úrovni desetinásobku. Jsou to různé typy buněk, které zajišťují odlišné funkce. Patří sem oligodendrocyty tvořící myelinovou pochvu okolo axonu, astrocyty, což jsou buňky zajišťující výživu neuronů z kapilár, ependimální buňky a mikroglie.[4]

Zdrojem všech myšlenek i emocí je komunikace mezi neurony a mozkové vlny jsou produkovány elektrickými pulzy od shluků neuronů, které mezi sebou komunikují. Mozkové vlny se pak detekují za pomoci sensorů umístěných na povrchu hlavy. Mění se podle toho, co cítíme a co děláme – dominují-li pomalé vlny, pocitem je únava, naopak dominují-li vlny na vyšších frekvencích, je pociťováno napětí.

Rychlost mozkových vln měříme v Hertzech (cykly za vteřinu) a dále je rozdělujeme právě na základě frekvence. Vlny Delta (0,5–3 Hz) jsou nejpomalejší a zároveň mají největší amplitudu. Vznikají v hlubokém bezesném spánku a při meditaci. V tomto

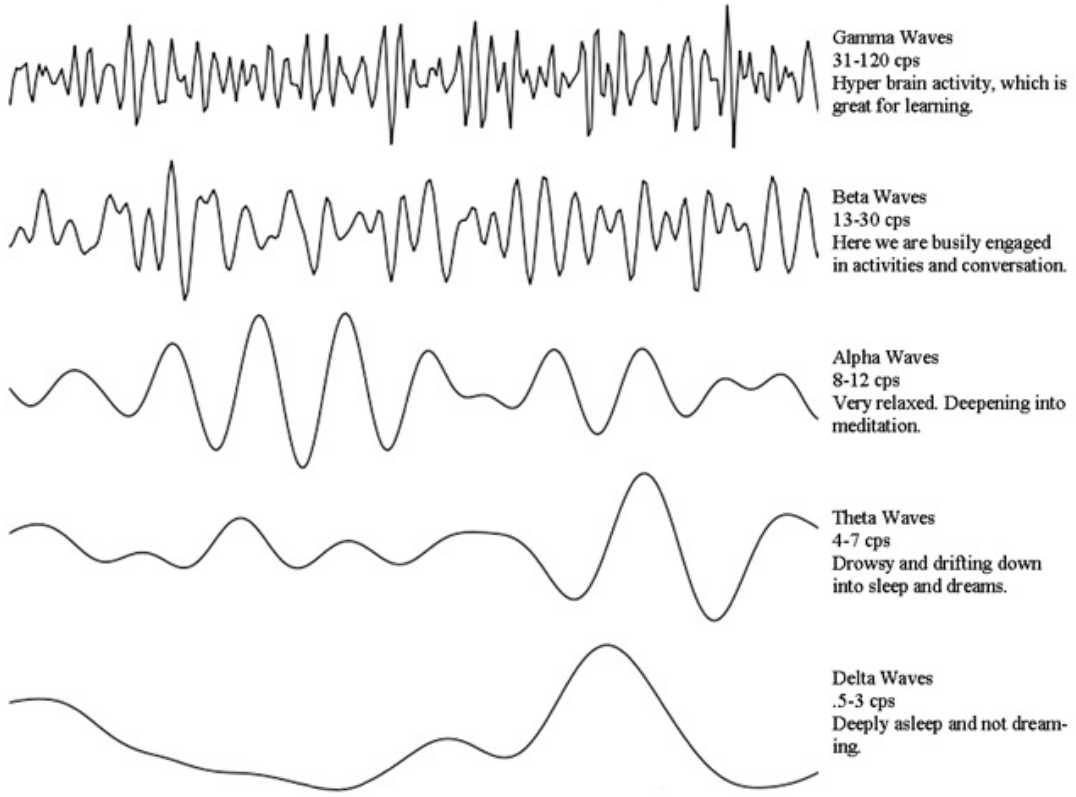
Obrázek 1 Struktura mozkové kůry[3]



stavu se také stimuluje regenerace a léčení. Vlny Theta (3–8 Hz) jsou opět vlnami typickými pro spánek a hlubokou meditaci. Tentokrát už se jedná o spánek se sny, mozek je zaměřen hlavně na signály pocházející zevnitř mozku (emoce). Alfa vlny (8–12 Hz) jsou přítomny při přemýšlení, pomáhají s koordinací, vyrovnaností a pozorností. Vlny Beta (12–38 Hz) dominují našemu stavu když jsme vzhůru, dáváme pozor a řešíme problémy. Tyto vlny dlouhodobě vyžadují velké množství energie. Gama vlny (38–42 Hz) jsou nejrychlejší vlny našeho mozku. Dlouhou dobu byly považovány pouze za šum, nicméně další výzkumy prokázaly jejich přítomnost při zažívání principů vyšších hodnot, altruismu, avšak jejich zdroj je záhadou, protože jejich frekvence přesahuje frekvenci generovanou neurony.

Mozkové vlny mají zásadní vliv na náš život. Jejich disbalance ovlivňuje negativně například kvalitu spánku, způsobuje agresivitu a deprese. Tyto disbalance jsou také typické pro úzkostné poruchy nebo ADHD. Mozkové vlny lze ovlivňovat buď chemickou cestou formou léků nebo drog, formou meditace popřípadě formou neurofeedbacku.[5] Vlny včetně frekvencí jsou znázorněny na obrázku 2.

Obrázek 2 Mozkové vlny v jednotlivých pásmech[5]



2.2 Elektroencefalografie (EEG)

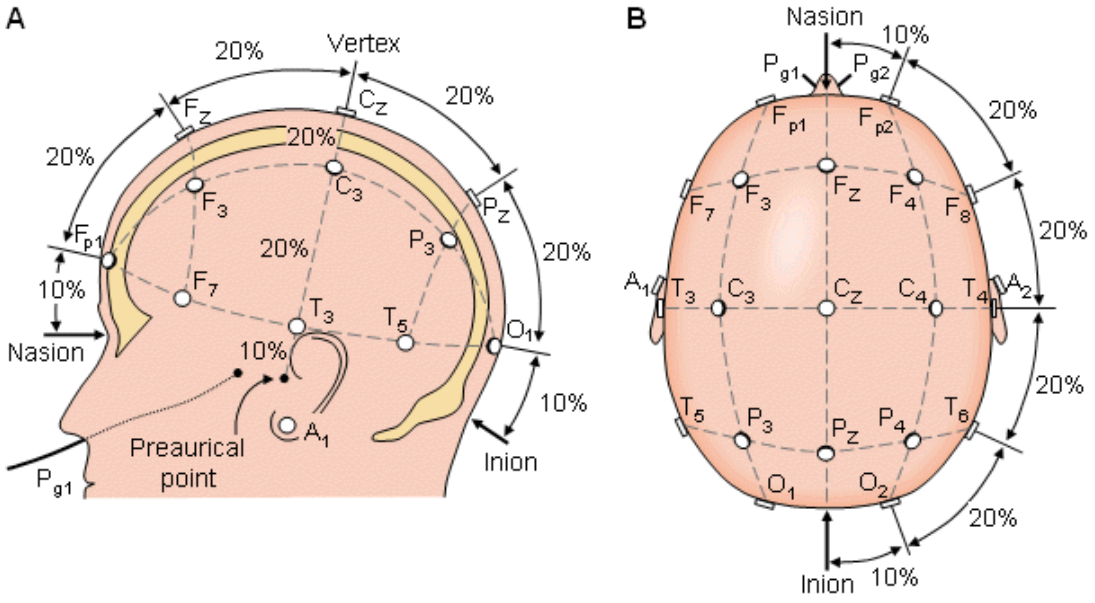
Elektroencefalograf představuje neinvazivní metodu nahrávání elektrické aktivity mozku. Nahrává se spontánní elektrická aktivita mozku za určitý čas, kdy jsou informace získávány ze senzorů umístěných na povrchu hlavy. Tato metoda je často užívána při diagnostice epilepsie, kómatu, mozkové smrti nebo poruch spánku. Historicky byl EEG využíván také pro diagnostikování nádorů nebo mrtvice, což se neosvědčilo, a proto bylo jeho používání nahrazeno hlavně moderními metodami jako je CT[1] nebo magnetická rezonance. Při praktickém využití EEG vyšetření trvá 20–30 minut a zahrnuje nahrávání analýzu dat získaných z elektrod zařízení.Také se používá při monitorování anestezie a účinku sedativ. Používá se i pro výzkumné účely v oblastech jako jsou kognitivní psychologie, neurolingvistika a psychofyziologie.

Mezi hlavní výhody využití patří nízké náklady, tichý chod, který umožňuje souběžně využít zvukové stimuly a neinvazivnost. Nevýhodou je rozlišení, protože není možné přesně určit, které části mozku jsou aktivovány. Zkoumá se právě spontánní aktivita, aktivita v reakci na podněty – zvukové nebo zrakové.

Informace o elektrické aktivitě mozku a možnostech jejího měření se datují do roku 1890, kdy byly publikovány lékařem Richardem Catonem. Další jméno spojené s podobným objevem při výzkumu prováděném na zvířatech je Adolf Beck, který ve stejném roce přišel s podobným poznatkem. V letech 1912 a 1914 se setkáváme s prvními měřeními, provedenými na zvířatech. První lidské EEG bylo naměřeno Hansem Bergerem v roce 1924. Dále v roce 1936 byla otevřena první laboratoř pro EEG ve státě Massachusetts. V dnešní době je EEG důležitým nástrojem právě pro neurofeedback.[6]

[1] počítačová tomografie

Obrázek 3 Umístění elektrod při snímání EEG[7]



Mezinárodní standard nahrávání EEG (označený 10-20) se využívá k nahrávání spontánní elektrické aktivity mozku. Na povrch hlavy se připevňuje 21 elektrod (viz obrázek 3). Pozice se určuje od dvou výchozích bodů. Na jedné straně se jedná o Nasion, prohlubeň u kořene nosu (mezi očima) a na druhé o Inion, kostnatý výběžek lebky v týle. Z těchto bodů jsou potom v intervalech 10 a 20 % umísťovány elektrody. Kromě tohoto standardního modelu existuje více systémů pro nahrávání EEG, například formát „Queen square system“ používaný pro měření reakce mozku na podněty.

2.3 Definice neurofeedbacku

Jako hlavní zdroj byla použita kniha Roberta Cobena a Jamese Evanse[8], dále pak internetové zdroje citované u konkrétních bodů. Pokud bychom hledali přesnou definici, tak v literatuře najdeme například tuto: „Neurofeedback je technikou, ve které cvičíme svůj mozek, aby pomáhal vylepšovat svoji schopnost regulovat veškeré tělesné funkce a postarat se o sebe“[9]. Mozek, který je trénován pomocí zpětné vazby, obecně může vylepšit spánkové vzorce, pomoci s úzkostí, depresemi a syndromy migrény popřípadě chronickými bolestmi hlavy. Dále také vylepšuje schopnost soustředit se a pomáhá s kontrolou emocí.

Dalšími oblastmi, kde je neurofeedback využíván je léčba epilepsie, různých typů zranění mozku, autismu či léčba poruch soustředění u dětí. Jedná se o dlouhodobou terapii složenou z pravidelných sezení, která probíhají v pravidelných intervalech. Neurofeedback zahrnuje nahrávání, analýzu a interpretaci výsledků kvantitativního měření EEG signálů v reálném čase za účelem sledování změn v elektrické aktivitě mozku. Výzkum v rámci tohoto oboru se zaměřuje na prediktivní algoritmy, které kombinují analýzu chování, neurokognitivní hodnocení a neurofyziologické metody.

Tato metoda je používána například při léčení symptomů ADHD a přináší zajímavé výsledky. V rámci neurofeedbacku se sledují EEG vlny a reakce mozku na různé podněty. Pokud se jedinec naučí tyto reakce ovládat, dochází ke zlepšení soustředění, výkonnosti a také k omezení projevů poruch v chování.

Další skupiny, u kterých je neurofeedback využíván, jsou pak lidé s autismem a jeho různými variantami. Neurofeedback využívá pokročilé počítačové technologie za účelem regulace mozkových vln. Vzniká informace o mozkových vlnách, která je potom konvertována do formy hry (vizuální, zvukovou či kombinovanou formou). Propojení počítače s mozkem je formou elektrod snímající mozkové vlny. Terapeut obratem dostává informace o tom, na jakých frekvencích se skutečně nachází mozkové vlny pacienta vůči tomu, kde by se nacházet měly. Na základě rozdílů se potom jedinec učí tyto diskrepance ovládat a regulovat. Cílem her je, naučit pacienta ovládat a vylepšovat strukturu jeho mozkových vln.

Výhodou neurofeedbacku je, že se jedná o neinvazivní léčbu bez negativních vedlejších účinků. Je důležité vzít v potaz také dlouhodobou aplikaci, která je zde nutná – léčba může trvat i několik měsíců. Stále je však časově méně náročná oproti behaviorální terapii, jejíž délka dosahuje i několika let. Další velkou výhodou je také to, že výsledek léčby přetrvává i po jejím ukončení. Nedochází tak ke znovuobnovení příznaků po vysazení, což se stává u léčby formou léků nebo dietou.

Výzkum také prokázal u léčení ADHD poruchy, že neurofeedback vylepšuje pozornost, inteligenční kvocient a pomáhá při problémech s impulzivností a hyperaktivitou. V posledních letech proběhlo několik extenzivních studií na toto téma, kdy byly sledovány skupiny jednotlivců postižených těmito chorobami a celkově byla zjištěna poměrně vysoká účinnost této léčby. [10] [8]

Další poruchou, u které se používá neurofeedback je epilepsie. Výzkum možnosti léčení epilepsie formou neurofeedbacku sahá až do šedesátých let minulého století. Neurofeedback zde umožňuje snížit frekvenci jednotlivých záchvatů i u pacientů, kterým nepomohla medikace. Nicméně v oblasti léčby epilepsie představuje neurofeedback stále spíše alternativní formu léčby.

Co se týče historie, tak neurofeedback byl poprvé popularizován v šedesátých letech minulého století psychologem Joem Kamiyou, v různých formách provází léčbu mnoha poruch do dnešního dne, kdy nabývá na významu a využití díky existenci pokročilých technologií.

3 Návrh GUI pro biofeedbackový systém

Tato kapitola je věnována základním informacím o návrhu aplikace. Jsou zde zmíněny důvody, které k tomuto návrhu vedly a je popsána základní struktura nové modulární architektury.

3.1 Historie a výchozí stav jádra aplikace

Na počátku byla malá aplikace, která uměla propojit zařízení na snímání mozkových vln s audio-vizuálním výstupem v podobě hry zobrazované na počítačové obrazovce. Jádro aplikace umělo zpracovávat vstupní data ze zařízení, analyzovat je a na základě provedené analýzy ovládat hru.

Aplikace sice byla funkční, ale architektonicky byla velmi jednoduchá a její rozšiřitelnost byla značně omezená. Došlo proto k zásadní úpravě vnitřní struktury výpočetního jádra a podle nového a robustnějšího návrhu byla velká část kódu přepsána. Během této pokročilé refaktorizace byla částečně obohacena funkcionalita jádra, došlo ale k zániku a odstranění grafického uživatelského rozhraní.

Aplikace obsahovala několik předprogramovaných testovacích scénářů umožňujících její spouštění ve vybraných režimech nad testovacími daty, jakékoliv úpravy však byly složité a zdlouhavé.

Jádro tedy poskytuje většinu potřebných funkcí pro připojování vstupních zařízení, přijímání a analytické zpracování vstupních dat ze zařízení, nastavování parametrů analýzy a částečně také spouštění a ovládání her.

3.2 Existující aplikace – průzkum trhu

Výsledná aplikace by měla zaplnit volné místo na trhu. Není cílem vytvořit další z řady jednoduchých (i když vizuálně často velmi zajímavých) zobrazovacích biofeedbackových aplikací.

Jako příklad takových aplikací je možno uvést produkty od firmy Pro-Spiro [11]. Jejich aplikace Biograph Infinity je schopná spouštět různé interaktivní zobrazení a obsahuje možnosti pokročilé vizualizace přijímaných dat.

Dalším příkladem jsou softwary od firem Neurosoft [12] a Mitsar-medical [13]. Shodně obsahují velké množství možností a způsobů zobrazování snímaných signálů spolu s různým nastavením parametrů, opět jim však chybí administrativní část.

Námi vyvíjený software je cílen na terapeutická centra obstarávající terapie velkému množství klientů. Taková centra potřebují přehlednou a jednoduchou správu klientů, možnost ukládat rozličná data pro jednotlivé klienty a sledovat vývoj výsledků klientů v průběhu terapií. Důraz je proto kladen na vytvoření funkčního spojení mezi administrativní aplikací pro správu dat terapií a klientů, aplikací pro zobrazování snímaných EEG dat a biofeedbackovou aplikací používající výsledky analýzy snímaných dat pro interakci s audio-vizuálními pomůckami (hrami, chytrými roboty) v rámci terapeutických sezení s pacienty.

3.3 Deklarace záměru

Cílem práce je vytvoření komplexního grafického uživatelského rozhraní nad stávajícím jádrem tak, aby výsledná aplikace umožňovala terapeutům snadno ovládat všechny potřebné funkce jádra. Grafické rozhraní musí být přehledné, intuitivní, snadno přizpůsobitelné potřebám jednotlivých terapeutů a lehce rozšiřovatelné v případě přidávání nové funkcionality jádra. Aplikace by měla přinášet harmonické spojení dynamického designu moderních aplikací s jednoduchostí a přehledností klasických kancelářských programů.

V první fázi vývoje se jedná o aplikaci pro stolní počítače, následně se však počítá i s dalšími variantami pro mobilní zařízení (tablety) nebo webové rozhraní pro vzdálený přístup. Návrh architektury s těmito fakty počítá a je připraven pro budoucí rozvoj.

3.4 Požadavky na aplikaci

Celý projekt lze efektivně rozdělit do tří částí, které spolu sice spolupracují, ale jinak jsou na sobě vlastně nezávislé. Jedná se o administrativní část, která umožňuje správu klientů, terapií, sezení a jejich jednotlivých kol, terapeutickou část, ve které jsou spravována jednotlivá zařízení, parametry počítaných analýz a spouštěny hry v rámci kol sezení a výkonnou část, která zajišťuje komunikaci s datovým úložištěm a kde se shromažďují data ze vstupních zařízení, probíhá jejich zpracování a jsou odesílána na zobrazovací zařízení.

3.4.1 Administrativní část

V rámci administrativní části je potřeba řídit přístup jednotlivých uživatelů k systému. Aplikace bude používána více terapeuty a každý z nich potřebuje jednoduchý přístup k datům svých klientů.

Po přihlášení se do systému pak bude terapeut moci vyhledávat a upravovat existující klienty a vytvářet nové. U každého klienta je potřeba evidovat jeho základní osobní údaje, jeho diagnózy, jednotlivé terapie, kterých se klient zůčastnil nebo právě účastní, a přehled sezení k jednolivým terapiím. Terapeut může měnit, přidávat nebo odebírat klientům diagnózy a přidávat a měnit terapie. Pro jednotlivé terapie může terapeut vytvářet nová sezení s klienty.

V rámci sezení pak terapeut vybírá z nabídky variant nastavení kanálů a zařízení podle terapie, ve které sezení probíhá, a spouští jednotlivá kola sezení.

Během kol jsou spouštěny hry, nahrávány signály ze vstupního zařízení a výsledky zobrazovány a ukládány do datového úložiště.

3.4.2 Terapeutická část

Terapeutická část umožňuje výběr vstupního zařízení, které bude během kol sezení používáno jako zdroj signálů pro analýzu. Jakmile je vybráno zařízení, je možné nastavovat jednolivé parametry analýzy a je možné spouštět hry.

Pro spuštění hry je potřeba vybrat jednu z dostupných her, zvolit na kterých všech výstupních zařízeních má být spuštěna. Ve chvíli, kdy jsou všechna vybraná výstupní zařízení připravena, lze spustit hru a tím začít jedno kolo sezení.

3.4.3 Výkonná část

Jedná se vlastně o původní jádro aplikace, nad kterým je stavěno uživatelské rozhraní. Obsahuje metody na připojování různých vstupních zařízení a čtení dat z nich, provádí matematické rozbory signálů, jejich analýzu podle nastavených parametrů a výsledky používá jako data pro tvorbu grafů v grafickém rozhraní a zároveň pro ovládání her.

Zajišťuje také kompletní komunikaci s datovým úložištěm a s ostatními komponentami aplikace – například s mobilním zařízením (tabletem) zajišťujícím připojení chytrých robotů.

3.5 Návrh architektury

Jako základní kámen celého grafického uživatelského prostředí byla zvolena platforma NetBeans. Ta se sama o sobě stará o několik nízkoúrovňových vlastností a poskytuje tak vhodné prostředí pro tvorbu dynamického uživatelského rozhraní.

Nejdůležitějšími vlastnostmi designu, na které byl při návrhu achitektury obzvláště kladen důraz, jsou dynamika rozhraní, modularita, robustnost a snadná rozšiřitelnost. Dynamika rozhraní a modularita jsou zajištěny díky použití platformy NetBeans. Každý jednotlivý kus uživatelského rozhraní zajišťující určitou funkcionalitu může být samostatným modulem v rámci plaformy a tyto moduly se pak v případě potřeby dají nahradit jiným modulem se stejnou funkcionalitou. Navíc platforma přináší možnost moduly přímo za běhu aplikace přeskupovat, přemísťovat nebo je zavírat a spouštět. Každý uživatel si tak může svoje pracovní prostředí přizpůsobit svým potřebám a preferencím.

Zároveň přísná modularita celého návrhu přináší příslib snadného budoucího vývoje a rozvoje nových funkcí. Bude-li do aplikace přidávána nová funkcionalita, stačí vytvořit nový modul zajišťující její fungování a téměř všechny ostatní existující moduly mohou zůstat v netknuté podobě.

3.5.1 Intermodulární komunikační protokol

Jednotlivé moduly aplikace mezi sebou samozřejmě potřebují komunikovat, potřebují se navzájem informovat o změnách svých stavů a potřebují mezi sebou sdílet data. Bylo proto potřeba navrhnout robustní intermodulární komunikační protokol.

Platforma NetBeans má v sobě zabudovaný systém, kterým mezi sebou mohou jednotlivé moduly sdílet objekty. Nad tímto systémem jsem navrhl zprávový protokol, který umožňuje velmi jednoduše používat funkcionalitu ostatních modulů, kterou sdílí přes veřejné rozhraní.

Protokol je navržen na základě návrhového vzoru publish-subscribe. Modul při odesílání zprávy proto neřeší, jakému modulu zprávu posílá – jen obsah zprávy a jméno rozhraní poskytující kýženou funkcionalitu. Zpracování zprávy je pak na modulu (nebo klidně modulech), které implementují příslušné rozhraní. To nadále napomáhá jednoduchosti rozšiřování systému, protože nové moduly mohou využívat řadu již existujících rutin jako je například kontrola zalogování uživatele nebo připojení vstupního zařízení. Nový modul tak může na tyto události reagovat pouze implementováním příslušného rozhraní bez jakýchkoliv změn na již existujících modulech.

Komunikační protokol je součástí samostatného modulu, který obsahuje nástroje pro snadné používání protokolu a sdílená data pro ostatní moduly. Každý modul aplikace, který chce využívat komunikační protokol proto musí být na tomto modulu závislý.

3.5.2 Moduly

Modulů v aplikaci je aktuálně celkem 10. Zde je stručný přehled.

- Bus modul – zajišťuje funkcionalitu intermodulárního komunikačního protokolu, obsahuje potřebné nástroje pro jeho používání a zároveň obsahuje sdílená data (rozhraní, entitní třídy a DTO třídy)
- Hub modul – původní jádro aplikace. Je v něm veškerá výpočetní funkcionalita, stará se o komunikaci s datovým úložištěm, sbírá data ze vstupních zařízení a ovládá hry.
- UserControl – přihlašování a odhlašování uživatelů
- ClientControl – správa klientů, diagnóz, terapií, sezení a kol
- DeviceSettings – správa vstupních zařízení
- FourierSettings – podrobná nastavení parametrů prováděných analýz
- SignalDisplay – grafické zobrazování signálů ze vstupních zařízení
- FourierDisplay – grafické zobrazování výsledků analýzy
- GameControl – nastavování parametrů a spouštění her
- TabletControl – připojování mobilních zařízení (tabletů)

Při popisu aplikace v návrhových a implementačních kapitolách budu postupovat právě podle jednotlivých modulů.

4 Technologie

V této kapitole se věnuji použitým technologiím. U každé je uveden její krátký popis, připomenuty její hlavní výhody a je vysvětleno, proč byla pro projekt vybrána.

4.1 Java

Java je multiplatformní objektově orientovaný programovací jazyk. V dnešním světe se používá doslova všude – od integrovaných čipů, přes mobilní zařízení, smartphone platformu Android, desktopové aplikace až po webové servery. V následujícím textu vycházím z veřejně dostupných informací. [14] [15]

Hlavními důvody obliby Javy jsou:

- jednoduchost – Jazyk je koncipován tak, aby byl syntakticky co nejjednodušší, kódy v něm psané snadno čitelné a byl celkově příjemný na práci.
- bezpečnost – Programy jsou spouštěny v prostředí virtuálního stroje, čímž se minimalizují možnosti výskytu chyb, které bývají časté při přímé práci s pamětí počítače. Chrání také systém před nebezpečnými operacemi nebo před napadením operačního systému nepřátelským kódem.
- silná typová kontrola – Každá používaná proměnná, musí mít svůj datový typ.
- objektový přístup – Každá proměnná, pokud se nejedná o jeden z osmi primitivních datových typů, je objektem.
- přenositelnost – Programy v Javě jsou nezávislé na platformě a lze je proto snadno přenášet mezi jednotlivými operačními systémy. To poskytuje mnohem širší pole působnosti na trhu, protože uživatelé nejsou limitováni požadavky pro běh aplikace.
- plná podpora vícevláknových aplikací
- inteligentní vývojové prostředí – Díky tomu, že je Java interpretovaný jazyk, mnoho chyb ve zdrojovém kódu lze odhalit ještě před samotnou kompilací programu a vývojové prostředí na ně programátora upozorní. Díky tomu je programování mnohem snazší a příjemnější.

Java byla použita pro vývoj původního jádra aplikace a proto i uživatelské rozhraní je vytvořeno v ní. Java je s její knihovnou Swing ideální pro tvorbu robustního grafického uživatelského rozhraní (GUI). Navíc díky použitému vývojovému prostředí NetBeans obsahujícím několikrát oceněný grafický designér Matisse byl návrh mnoha částí GUI zásadně rychlejší a pohodlnější než jakékoliv jiné možnosti.

4.2 Platforma NetBeans

NetBeans platforma [16] je generický framework pro tvorbu Swingových aplikací v jazyce Java. Poskytuje vývojáři základní nízkoúrovňovou funkcionalitu, která by jinak zabrala obrovské množství času v rámci vývoje.

Nejdůležitejší vlastností je její kompletní modularita. Výsledná aplikace se skládá z množství modulů, které mezi sebou samozřejmě mohou komunikovat, spolupracovat a sdílet data. Prvoplánově jsou však stavěné tak, aby byly na sobě nezávislé. Platforma

pak přináší propracovaný systém rozmísťování modulů v rámci aplikace, měnení jejich pozice a velikosti za běhu aplikace, možnost jejich minimalizace do postraních lišt nebo jejich úplné zavření s možností je později znovu spustit z nabídky. Je tak věcí každého uživatele, aby si pracovní prostředí prizpůsobil svému vkusu a svým potřebám.

Lze také používat perspektivy (přednastavená rozložení modulů) a přepínat mezi nimi. Lze tak efektivně během provádění různých činností měnit celé rozhraní podle toho, co je zrovna potřeba zobrazovat.

Platforma dále zajišťuje snadnou a pohodlnou distribuci výsledného softwaru v podobě instalátoru. Instalátor je sice sestaven přímo pro určitý operační systém, ale dají se generovat instalátory pro většinu dostupných operačních systémů. Instalátor obsahuje standardizovaného instalačního průvodce, který může být dále rozšiřován pro potřeby aplikace. V základní verzi umožní uživateli vybrat umístění aplikace, nabídne vytvoření zástupců v obvyklých umístěních (startovní nabídka, pracovní plocha) a zároveň v systému vytvoří uninstaller pro snadné odebrání aplikace.

Změny v aplikaci mohou být distribuovány jako malé instalační balíčky pro jednotlivé moduly. Platforma dokonce obsahuje vlastní systém auto-update serveru, kdy aplikace po spuštění sama zkontroluje, zda nejsou dostupné nové verze některých jejích modulů a pokud jsou, stáhne je a nainstaluje. Je tak velmi snadné držet veškeré běžící aplikace vždy aktualizované a zásadním způsobem to zkracuje čas mezi opravením chyby programátory a nainstalováním aktualizace na běžících aplikacích.

Samozřejmostí platformy je pohodlná podpora vícejazyčnosti celého rozhraní.

4.3 Databáze MySQL

MySQL [17] je sada databázových strojů vyvíjených společností Sun Microsystems (dnes spadající pod Oracle Corporation) poskytujících šikovné řešení doslova pro každý projekt. Na MySQL dnes spoléhá obrovské množství nadnárodních organizací. Dokonce 9 z 10 největších světových webů[1], mezi nimi například Facebook, Google a YouTube běží práve na MySQL. [18]

Přestože je napsána v jazyce C++, je dostupná pro širokou nabídku platforem. Jsou dostupné verze pro všechny Windows, celou řadu systému na bázi Linuxu i pro MacOS. MySQL nabízí všechno od vestavěných databází, přes klasické databázové stroje až po clusterové databáze. Díky dvojímu licencování je možné pro využívat buď zdarma poskytovanou komunitní verzi MySQL pod GPL licencí nebo MySQL Enterprise, která k samotné databázi přidává širokou řadu nástrojů a služeb.

Po dlouhou dobu byla MySQL vytýkána nižší výkonnost pro větší projekty. Od představení technologie InnoDB, která nahradila do té doby používanou MyISAM, tuto skutečnost částečně eliminovala. Dalším vývojem pak mezi verzemi 5.1 a 5.5 došlo ke zrychlení o neuvěřitelných 1500

Největší výhodou MySQL a základním důvodem její obliby je ale bezesporu její jednoduchost a nenáročnost co se týče administrace. Instalace je připravená v několika různých balíčcích podle toho, co je na projekt zrovna potřeba a po ukončení instalačního průvodce je databázový stroj běžíci, nakonfigurovaný a připravený k okamžitému použití bez jakýchkoliv dalších zásahů. [18]

Pro MySQL také existuje řada administračních nástrojů ještě dále usnadňující správu databází i databázového stroje. Nejznámějšími a nejrozšířenějšími jsou PHPMyAdmin a jeho mladší sourozenec Adminer, webové nástroje na bázi PHP, nebo MySQLWorkbench.

[1] Podle seznamu Alexa a jejich top 500 webových stránek http://www.alexa.com/topsites/global;0

Jako úložiště dat pro projekt byl zvolen plnohodnotný standalone databázový stroj MySQL zdarma pod GPL licencí. Díky výše zmíněným výhodám a v kombinaci s JPA se ukázal jako ideální řešení.

4.4 Java Persistence API (JPA)

JPA je framework od společnosti Oracle umožňující objektově relační mapování pro standardní Javu i její Enterprise variantu. Je tedy mezivrstvou mezi aplikací a její databází. JPA podporuje širokou řadu standardizovaných SQL datových typů proměnných a všechny dostupné vazby mezi jednotlivými entitami.

Pro JPA je potřeba vytvořit anotované entitní Java třídy, kde anotace obstarávají správné namapování dat z databázové tabulky na daný objekt. Práce s namapovanými objekty pak už je velmi přímočará a jednoduchá.

JPA samo o sobě je pouze rozhraním, vnitřně pak používá jednu z technologií Hibernate, Oracle Toplink nebo OpenJPA. V projektu je jako implementace použito OpenJPA.

4.5 Universal Modelling Language (UML)

UML je jazyk pro modelování návrhů objektově orientovaného softwarového inženýrství. Tento standard byl vytvořen konsorciem Object Management Group (OMG). Do UML spadají notace grafického zápisu, které vytváří vizuální modely pro objektově orientované systémy. Tento jazyk se používá ke specifikaci, vizualizaci, konstrukci a dokumentaci částí vyvíjeného objektově orientovaného softwaru. UML definuje základní stavební prvky, ze kterých pak skládá diagramy – aktéry, databázová schémata, případy užití, znovupoužitelné softwarové komponenty a další. Kombinuje techniky datového, business, objektového a komponentového modelování. Tento modelovací jazyk spojuje notace Boochovy metody, techniky objektového modelování (OMT) a objektově orientovaného softwarového inženýrství (OOSE).

Z diagramů vytvořených v UML se dají generovat části (nebo dokonce funkční celky) přímo v jazyce implementace – například převedení diagramu tříd do objektů jazyku Java nebo vygenerování databázového skriptu z databázového modelu. UML nabízí dva mechanismy pro úpravy – profily a stereotypy.

UML modely se dělí do dvou skupin. První jsou statické (strukturální) modely, které podtrhují statickou strukturu systému za použití objektů, atributů, operací a vztahů. Do této části patří diagramy tříd, diagramy nasazení a diagramy komponent. Dynamické (behaviorální) modely podtrhují dynamické charakteristiky systému poukazováním na interakci objektů a jejich vnitřní změny. Do této části lze zařadit mezi jinými například sekvenční diagramy a diagram aktivit.

Pro jeho použití jsem se rozhodl pro jeho přehlednost a názornost v něm vytvořených modelů. Je to jeden z nejefektivnějších způsobů, jak rychle, přesně a pohodlně vytvořit diagramy pro popis různých částí systému i celého systému z různých úhlů pohledu.

5 Analytická dokumentace

V této kapitole je uveden a podrobně rozebrán konceptuální datový model databáze. Pro každý z modulů aplikace je zde popsáno jeho očekávané chování a je vysvětleno, jak spolu moduly spolupracují.

5.1 Databázové entity

Na obrázku 4 jsou vidět vztahy mezi jednotlivými entitami, které je nutné ukládat v databázi. Následuje popis jednotlivých entit.

5.1.1 Personál

Osoba používající aplikaci. Je potřeba evidovat jméno a heslo pro potřeby přihlašování do systému, její přístupová práva v rámci systému a zda-li nebyla deaktivována.

Ačkoli personál nemá žádné přímé spojení s klienty, je s nimi svázán přes terapie – každá terapie má vedoucího terapeuta. Když jsou po přihlášení personálu do systému zobrazováni klienti, přihlášenému uživateli se jako „jeho“ klienti zobrazují tací, u kterých je terapeutem alespoň v jedné terapii.

5.1.2 Klient

Osoba navštěvující terapie. Ukládají se základní osobní a kontaktní údaje. Klientovi může být diagnostikováno více diagnóz a může se účastnit více terapií najednou. Je dokonce možné, aby s jedním terapeutem měl několik různých terapií.

5.1.3 Diagnóza a skupina diagnóz

Diagnózy a skupiny diagnóz jsou vytvořeny na základě MKN-10[1]. Každá diagnóza spadá právě do jedné skupiny a na základě skupin jsou pak pro rychlejší vyhledávání filtrovány při zobrazování v uživatelském rozhraní.

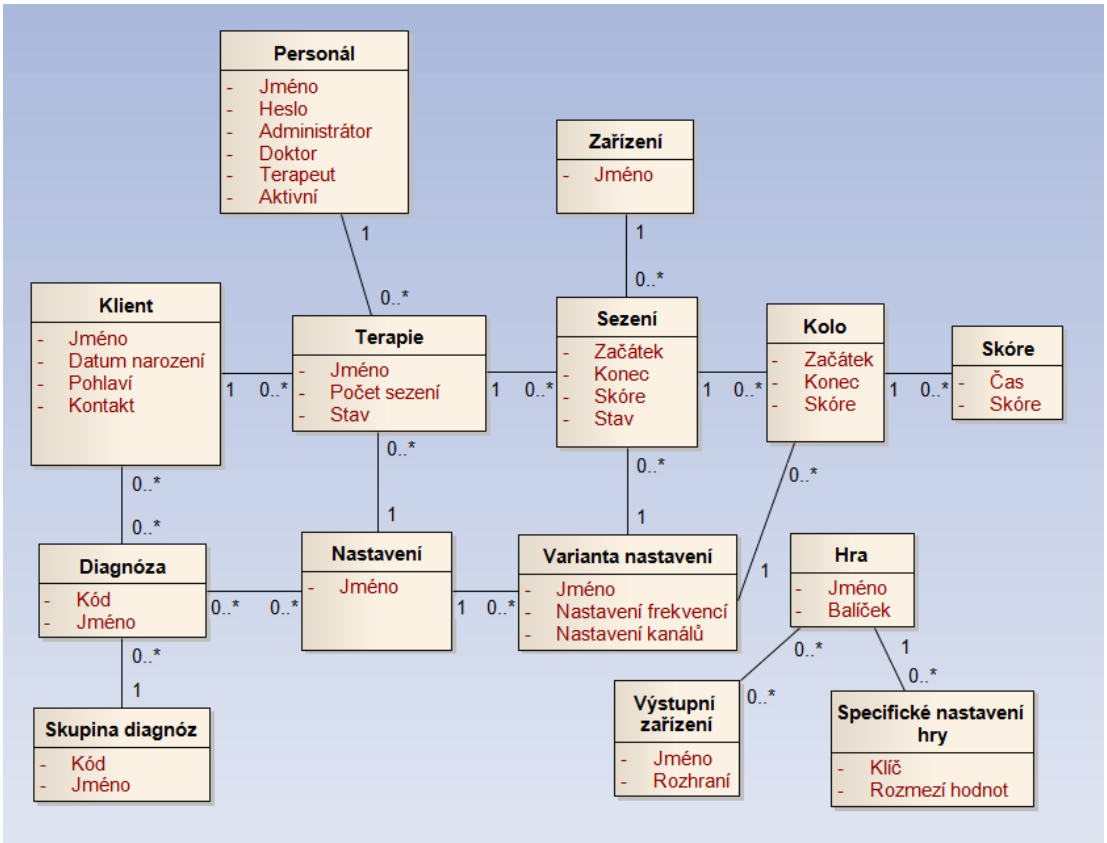
Diagnózy jsou také svázány s nastavením hodnot pro terapie. Různé diagnózy pro své terapie vyžadují odlišná nastavení parametrů a tudíž nelze nastavení libovolně zaměňovat.

5.1.4 Terapie

Terapie jsou bloky jednotlivých sezení pro daného klienta. Za každou je zodpovědný jeden člen personálu (terapeut) a každá terapie se váže k určité diagnóze přes nastavení hodnot parametrů terapie.

[1] Mezinárodní klasifikace nemocí a přidružených zdravotních problémů, 10. revize (MKN-10) se zapracováním změn, které obsahuje aktualizace mezinárodní verze International Classification of Diseases k 1. 1. 2014. http://www.uzis.cz/zpravy/upravena-verze-mkn-10

Obrázek 4 Návrh databáze – entitní vztahy



5.1.5 Zařízení

Vstupní zařízení, ze kterých aplikace přijímá data pro zpracování. Záznamy jednotlivých zařízení v databázi jsou pouze informativní názvy zařízení, které jsou přiřazovány k sezení. Slouží terapeutům pouze k identifikaci, na kterém zařízení sezení proběhlo a nemají žádnou vnitřní spojitost s typy zařízení, které se připojují do systému.

5.1.6 Nastavení a varianty nastavení

Nastavení je soubor hodnot parametrů, které se používají ve výpočtech v průběhu sezení.

Objekty nastavení slouží především jako propojení mezi terapiemi a diagnózami. Každá terapie má právě jedno nastavení, které obsahuje několik svých variant. Veškeré hodnoty parametrů jsou až ve variantách nastavení a jednotlivé varianty lze zjednodušeně chápat jako obtížnosti.

Ke každému sezení z terapie je přiřazena jedna varianta nastavení, která spadá pod nastavení zvolené v terapii onoho sezení.

5.1.7 Sezení, kola a skóre

Záznamy o jednotlivých schůzkách s klienty. Sezení jsou skupiny kol, které probíhají v určitém časovém rozmezí ohraničeném začátkem a koncem sezení. Skóre sezení je vypočítáno ze skóre jeho proběhlých kol při jeho ukončení. Stavy, ve kterých se sezení může nacházet, jsou probíhající, ukončené nebo zrušené.

Během nahrávání kola se do databáze ukládají záznamy o jeho průběhu jako objekty typu skóre. Po nahrání celého kola se z těchto hodnot vypočítá celkové skóre kola.

Sezení má přiřazenu variantu nastavení parametrů, nastavení kanálů se však v průběhu sezení může měnit, a tak je potřeba ukládat nastavení kanálů i pro jednotlivá kola. Při zpětné analýze je potřebné znát nastavení pro každé kolo, jen tak je možné správně interpretovat uložené hodnoty skóre.

5.1.8 Hry, specifická nastavení her a výstupní zařízení

Hry se spouštějí během jednotlivých kol a zaznamenávají se výsledky klienta. Hry mohou podporovat více výstupních zařízení. V případě, že hra vyžaduje jakákoliv další nastavení hodnot mimo běžné parametry, zde jsou uloženy ve formě párů jméno-hodnota.

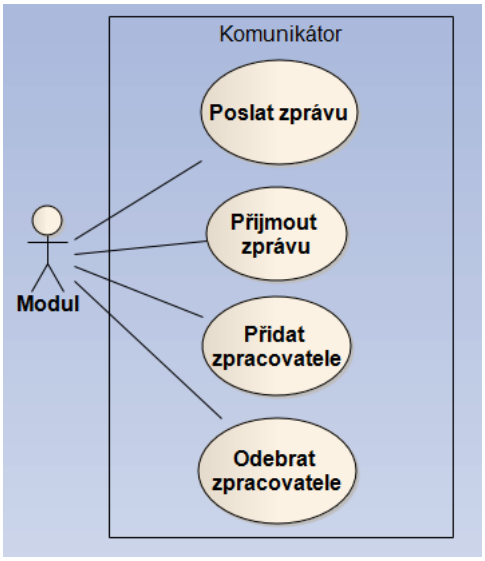
Výstupní zařízení, jsou rozhraní na které lze posílat výstupy ze hry. Typicky počítačová obrazovka, ale může to být třeba i inteligentní robot nebo mobilní zařízení (tablet). Výstupní zařízení jsou popsána jejich rozhraním, přes které s nimi pak GameControl modul komunikuje.

5.2 Popis jednotlivých modulů a jejich případů užití

5.2.1 Bus modul a intermodulární komunikační protokol

Bus modul je jedním z modulů, které nemají uživatelské rozhraní. Přesto je jeho existence naprosto esenciální pro fungování celého systému. Obsahuje v sobě veškeré potřebné prvky pro intermodulární komunikaci a zajišťuje přenos jednotlivých zpráv.

Obrázek 5 Případy užití komunikátoru



Jeho základními součástmi jsou sběrnice (Bus), komunikátor, zprávy a veřejná rozhraní.

Sběrnice přímo obaluje lookup systém NetBeans platformy a stará se tak přímo o posílání a přijímání zpráv. V rámci celého návrhu systému je sběrnice unikátní a vždy existuje jen jedna.

Komunikátor je pak konstrukce, která přímo využívá sběrnici. Každý modul si při svém vzniku vytvoří komunikátor a ten se automaticky připojí ke sběrnici. Od okamžiku vytvoření komunikátoru je modul schopen posílat a přijímat zprávy. Aby modul

mohl zpracovávat příchozí zprávy, je potřeba v komunikátoru přihlásit odběratele zpráv. Existují dva druhy odběratelů – vykonavatelé (Doers) a zpracovatelé (Processors). Pro přihlášení k odběru je zapotřebí poskytnout komunikátoru jméno rozhraní, pro které chce modul zpracovávat zprávy, a objekt implementující dané rozhraní.

Zpráva je přímo objekt, který je posílán komunikátorem přes protokol. Zprávy jsou jednoho ze šesti typů.

- Control – volání funkce rozhraní s poskytnutými parametry bez očekávání odpovědi
- Call – volání funkce rozhraní s poskytnutými parametry, návratová hodnota je odeslána zpět jako typ Callback
- Callback – odpověď na volání Call
- Data – zpráva obsahuje data na zpracování. Tento typ zprávy zpracovavá Processor, nikoli Doer
- Error – kritická chyba. V aktuální verzi nemá využití, existuje pro budoucí potřeby například při komunikaci se vzdáleným rozhraním po síti.
- Info – pouze informační zpráva pro systém jako takový, nikoli pro uživatele. Stejně jako u předchozího bodu, jeho využití je otázkou dalšího vývoje.

Rozhraní jsou v tomto modulu proto, že jsou na něm všechny ostatní moduly závislé. Jsou tak všechny sdílené věci na jednom místě a lze je pohodlně používat ve všech modulech. Mezi další sdílené věci patří různé konstanty, speciální grafické elementy uživatelského prostředí, nastavení fontů a obrázky.

5.2.2 Hub modul

Stejně jako Bus modul, ani Hub nemá grafické rozhraní. Svým způsobem jsou jeho grafickým rozhraním všechny ostatní moduly, Hub slouží jako výpočetní jádro.

Hub poskytuje ostatním modulům konektivitu k databázi a s ní spojené funkce na získávání dat, jejich tvoření a úpravu i jejich mazání. Dále přijímá a zpracovává data ze vstupních zařízení. Přijatá data ukládá do databáze a zároveň je analyticky zpracovává a výsledky analýzy posílá dalším modulům k vykreslení. Na základě přijatých dat pak také ovládá právě spuštěné hry.

5.2.3 UserControl

Jedná se o minimalistický modul, který řeší jen a pouze přihlašování a odhlašování uživatelů.

Systém uživateli nabídne seznam personálu, ten se vybere, vyplní heslo a zadaná data potvrdí. Pokud bylo heslo zadáno špatně, modul uživatele o této skutečnosti informuje chybovou hláškou. Jestliže heslo odpovídá, dojde k přihlášení uživatele, všechny moduly jsou o proběhlém přihlášení informovány.

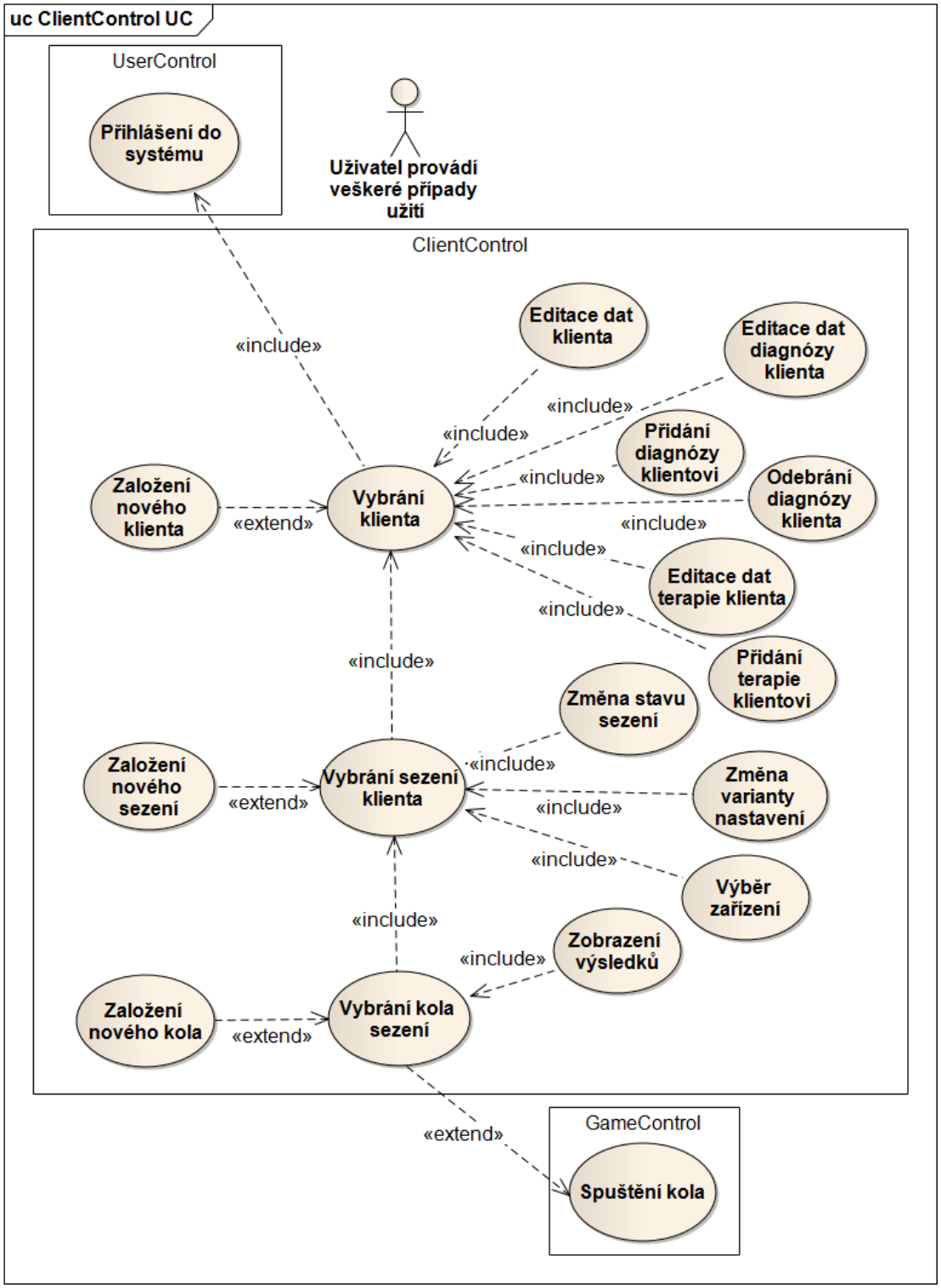
5.2.4 ClientControl

Administrativní modul, který má na starosti kompletní správu klientů, jejich diagnóz a terapií, sezení a kol.

Modul vyžaduje, aby v systému byl přihlášen uživatel. Pokud tomu tak není, modul místo běžné funkcionality zobrazí pouze informační hlášku, že není přihlášen uživatel.

Modul je rozdělen do tří sekcí – sekce správy klientů, sekce správy sezení a sekce správy kol.

Obrázek 6 Případy užití modulu ClientControl



V klientské sekci je seznam klientů s možností vyhledávání a po vybrání klienta ze seznamu pak jeho detaily. Před vybráním klienta je zobrazena pouze informační zpráva, že není vybrán žádný klient.

Detaily jsou rozdělené do čtyř částí.

V první jsou osobní údaje klienta jako jméno a příjmení, datum narození a poznámka terapeuta. Tyto údaje lze přepnout do režimu úprav. Úpravy dat lze potvrdit a tím uložit do databáze nebo změny zrušit.

V druhé části je filtrovatelný seznam sezení klienta. Sezení v seznamu jsou zobrazována chronologicky za sebou, nehledě na terapie, do kterých patří. Seznam lze filtrovat podle jednotlivých terapií klienta a to zapnutím filtrování a vybráním požadované terapie z poskytnutého seznamu. Ze seznamu sezení lze přejít přímo do sekce správy sezení na detail vybraného sezení.

Ve třetí části jsou k dispozici diagnózy klienta. Zobrazuje se seznam všech aktuálně přiřazených diagnóz včetně jejich skupiny. U každé existující diagnózy je možnost jí editovat nebo ji úplně odstranit. Terapeut může přidávat nové diagnózy. V případech přidání a editace diagnózy se oddíl diagnóz přepne do režimu úprav. V něm uživatel může vybrat ze seznamu skupinu diagnóz a diagnózu a v případě potřeby vyplnit dodatečné poznámky. Po potvrzení přidání nebo změn jsou data uložena do databáze a oddíl detailů se přepne zpátky na zobrazení celého seznamu diagnóz.

Poslední částí jsou terapie klienta. Stejně jako v předchozí části je zde přehledný seznam jednotlivých terapií klienta. Terapie narozdíl od diagnóz nejdou mazat, jdou pouze upravovat a lze přidávat nové terapie. Stejně jako u diagnóz se při přidávání nebo úpravě existující terapie přepne celý oddíl do režimu úprav a uživatel může vyplnit nebo změnit veškeré potřebné informace. Po potvrzení se data terapie uloží do databáze.

V druhé sekci modulu, správě sezení, je dostupný malý informativní blok s několika základními daty o právě vybraném klientovi, dále pak seznam sezení klienta, který stejně jako v předchozí části může být filtrován podle jednotlivých terapií. Přes informativní blok se lze pohodlně vrátit zpět na zobrazení sekce správy klientů a zobrazení detailů aktivního klienta. V hlavní části sekce se zobrazují detaily sezení, nebo dokud není vybráno sezení, tak pouze informační zpráva, že není vybráno žádné sezení.

Detaily sezení se skládají ze tří částí.

V první jsou informace o sezení – čas a datum jeho začátku a konce, terapie do které patří, jeho aktuální stav, celkové skóre sezení a vstupní zařízení, na kterém sezení probíhalo. V této části lze měnit stav sezení. Nově založená sezení mají automaticky stav „probíhající“. Další možné stavy jsou „ukončené“ a „zrušené“. Když je sezení přepnuto ze stavu „probíhající“, provede se uzavření sezení – spočítá se výsledné skóre sezení z výsledků jeho proběhlých kol a nastaví se datum a čas jeho uzavření. Rozdíl mezi stavy „ukončené“ a „zrušené“ je, že kola ve stavu „zrušené“ se nepromítají do průběhu terapie. Terapie a zařízení jsou hodnoty neměnné, nastavují se pouze při zakládání nového sezení.

Druhá část obsahuje informace o variantě nastavení pásem a kanálů pro vybrané sezení. Tuto část je možno přepnout do režimu editace a lze pak variantu nastavení měnit. Při vybírání nové varianty nastavení jsou zobrazována data pro právě vybranou variantu nastavení stejně jako v režimu zobrazení. Uživatel tak má možnost si před potvrzením nové varianty nastavení prohlédnout, jaká data nastavuje. Po potvrzení se část přepne zpátky do režimu zobrazení.

Ve třetí části je seznam jednotlivých kol sezení. O každém kole je v seznamu zobrazeno několik základních údajů a vybráním kola ze seznamu lze přejít přímo na zobrazení jeho detailů v sekci správy kol.

Poslední sekcí modulu je správa kol. Chováním velmi připomíná předchozí sekce, ale díky povaze entity kola je zásadně jednodušší.

Stejně jako v sekci správy sezení je zde informační blok s údaji aktuálně vybraného klienta, přes který lze snadno přejít přímo na zobrazení detailů klienta. Dále se zobrazuje informační blok s údaji o aktuálně vybraném sezení, pro které jsou zobrazena kola a pro které jsou zakládána nová kola a je možno přes něj přejít na zobrazení detailů sezení. Kola sezení jsou zobrazena ve stejně formátovaném seznamu jako v sekci správy sezení. Po vzoru předchozích sekcí, před vybráním kola ze seznamu je zobrazena pouze zpráva, informující uživatele, že má vybrat kolo ze seznamu nebo založit nové. Jakmile je kolo vybráno, zobrazí se jeho detaily.

Zobrazení detailů se liší podle stavu kola. Pokud je kolo nově vytvořené, nabídne aplikace uživateli spuštění kola a nahrání jeho průběhu. Pokud je kolo ukončené, má tedy nahraný svůj průběh, zobrazí se graf zobrazující vývoj skóre klienta v průběhu kola a uživateli je nabídnuta možnost založit a nahrát následující kolo.

Společnou částí všech sekcí je oblast, kterou jsem nazval Časová osa. Ta slouží jako jednoduchý a přehledný ukazatel aktuálního stavu administračního modulu a zároveň umožňuje rychlou a intuitivní navigaci mezi jednotlivými sekcemi.

Po přihlášení uživatele je aktivní pouze část „Klient“. Jakmile uživatel vybere nějakého klienta ze seznamu nebo založí nového klienta, zpřístupní se část „Sezení“. Stejně tak ve chvíli, kdy je vybráno v sekci správy klientů nebo správy sezení nějaké sezení, odemkne se část „Kolo“. Přes jednotlivé části časové osy lze okamžitě přejít do příslušné sekce modulu.

5.2.5 DeviceSettings

Modul pro připojování a ovládání vstupních zařízení, popřípadě spouštění signálů ze záznamu. Zároveň slouží pro nastavení, které kanály se mají zobrazovat při přímém vykreslování vstupních signálů.

Pro přímé připojení vstupního zařízení je potřeba ze seznamu vybrat jedno z dostupných zařízení a výběr potvrdit. Modul pak pošle požadavek na Hub, který požadované zařízení připojí a připraví se napřijímání signálu z něj.

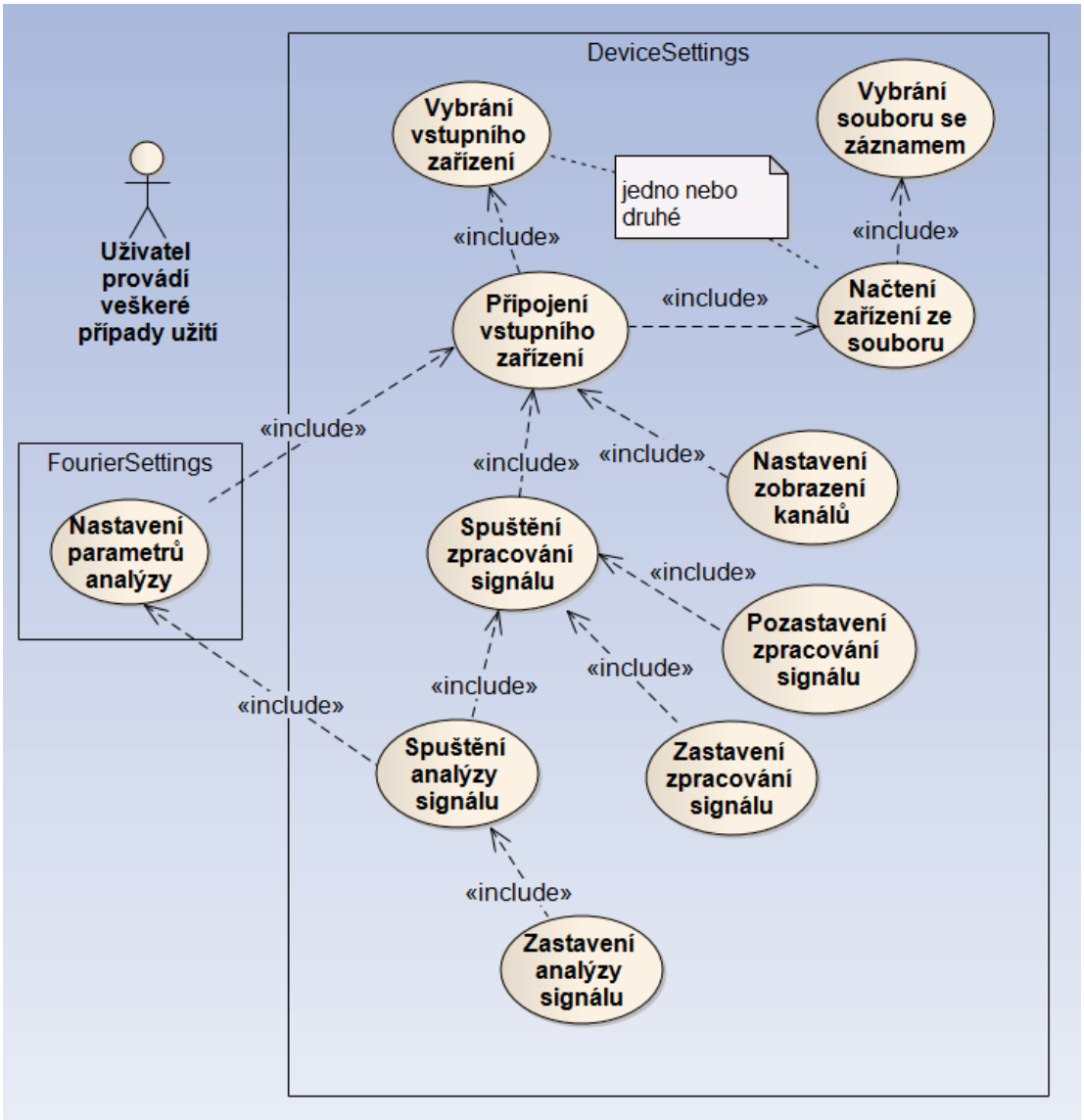
Při spouštění signálu ze záznamu je třeba nasměrovat modul k souboru obsahujícímu nahraná data. Modul se pak pokusí o připojení zařízení ze souboru a pokud soubor odpovídá hlavičkou i strukturou podporovaným formátům, modul vypíše informace o datech uložených v souboru. Pak je možné zařízení ze souboru připojit stejně, jako by se připojovalo jakékoliv jiné vstupní zařízení.

Modul se přepne do režimu připojeného zařízení a informuje ostatní moduly, že bylo připojeno zařízení. Dále je zobrazen přehled kanálů zařízení, kde je možné vybrat, které kanály zařízení se budou vykreslovat v SignalDisplay modulu. Zároveň se zpřístupní volba spuštění přijímání signálu z vybraného vstupního zařízení. Po spuštění čistých signálu se zpřístupní možnost spuštění analýzy dat a zapauzování nebo zastavení přijímání signálu. Při zapauzování se pouze pozastavní signál, při odpauzování se pokračuje ve zpracovávání v místě pozastavení. Při úplném zastavení dojde úplně k přerušení zpracovávání dat a k odpojení zařízení. Pro další zpracování dat signálu je potřeba znovu připojit nějaké vstupní zařízení buď přímo nebo ze souboru.

5.2.6 FourierSettings

Modul pro nastavování všech možností a parametrů pro výpočet Fourierovy transformace nad daty signálů ze vstupního zařízení. Pro nastavovnání parametrů modul

Obrázek 7 Případy užití modulu DeviceSettings

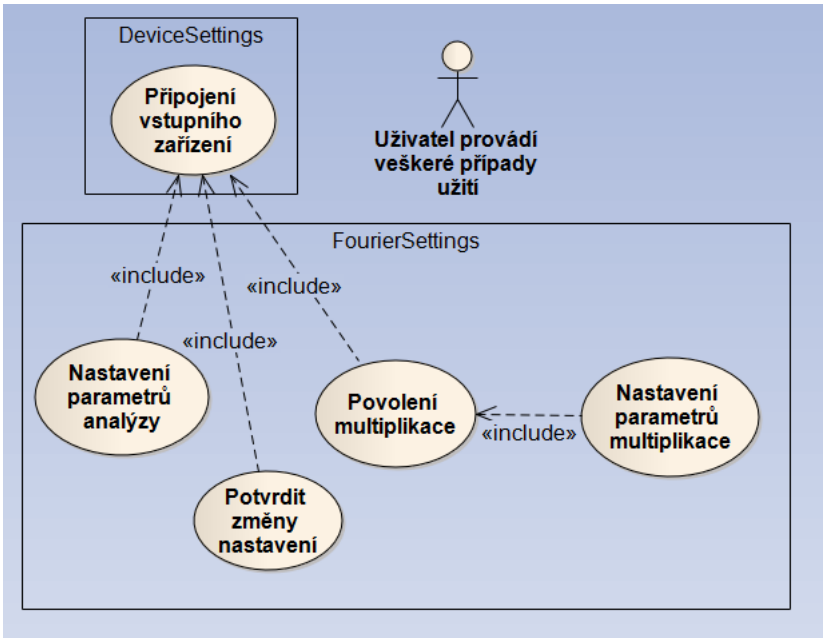


vyžaduje připojení vstupního zařízení přes modul DeviceSettings, dokud není vstupní zařízení připojeno, modul zobrazuje pouze informaci, že pro jeho funkci je potřeba připojit vstupní zařízení.

Stejně jako u zpracovávání přímých vstupních signálů lze i u transformace vybrat, které kanály vstupního zařízení se budou používat a dále pak zobrazovat v modulu FourierDisplay. Mezi nastavitelné parametry patří pokrytí, počet vzorků, typ a parametr okénka, rozsahy jednotlivých pásem a multiplikace. Multiplikace je ve výchozím stavu vypnutá, lze ji zapnout a pak jí nastavit typ a její tři parametry.

Ve výchozím stavu modulu se musí jednotlivé změny v nastavení parametrů manuálně potvrzovat. Uživatel může modul přepnout do stavu, kdy modul provedené změny automaticky potvrzuje v momentě, kdy je uživatel udělá. To je zvláště pohodlné ve chvíli, kdy je potřeba parametry transformace upravovat pouze jemně a přímo za běhu analýzy. Modul jde přepnout i zpět do výchozího stavu a nové změny se pak musí znovu explicitně potvrzovat.

Obrázek 8 Případy užití modulu FourierSettings



5.2.7 SignalDisplay

Zajišťuje vykreslování čistých signálů ze vstupního zařízení a informuje o kvalitě připojení jednotlivých elektrod na zařízení. Pro každý kanál je tak dostupný ukazatel rozlišující několik stavů elektrody.

Modul potřebuje pro své fungování připojené vstupní zařízení přes modul DeviceSettings. Pokud zařízení není připojeno, modul zobrazí informaci, že je potřeba nejprve nějaké zařízení připojit.

Jaké kanály se budou vykreslovat se nastavuje v modulu DeviceSettings.

5.2.8 FourierDisplay

Modul pro vykreslování výsledků analýzy. Pro každý kanál vstupního zařízení je vykreslen jeden graf s výsledky analýzy, navíc je speciální graf pro kombinované výsledky všech kanálů. Pro které kanály se zobrazují výsledky se nastavuje v modulu FourierSettings.

Modul potřebuje pro své fungování nastavené parametry analýzy z modulu FourierSettings. Pokud parametry ještě nebyly pro aktuálně připojené vstupní zařízení nastaveny, modul zobrazí informaci, že je potřeba nastavit parametry analýzy.

5.2.9 GameControl

Modul pro nastavování a spouštění her. Modul je rozdelen do čtyř částí.

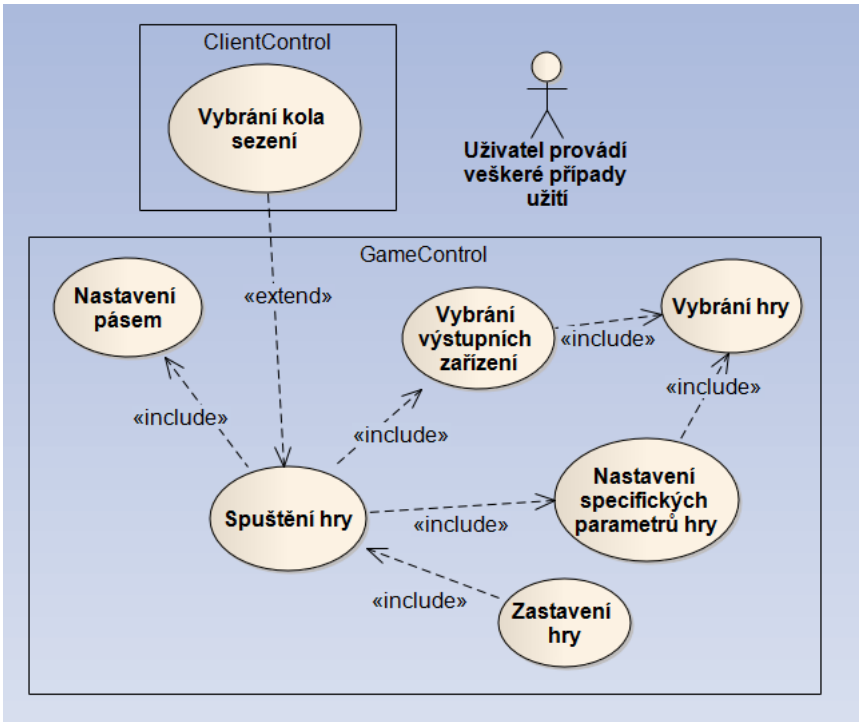
První částí je informační oblast, kde se zobrazí aktuálně otevřené kolo sezení v modulu ClientControl, pokud již není ukončené. Hry jdou spouštět i bez otevřeného kola, nicméně nedojde k nahrání a uložení výsledků do databáze.

V druhé části se nastavují parametry pro jednotlivá pásma zařízení. Hra poskytne modulu informace o počtu kanálů, které pro své fungování vyžaduje a na základě této informace je upraveno zobrazení nastavování parametrů.

Obsah třetí části se mění na základě vybrané hry ke spuštění. Pokud hra obsahuje nějaká specifická nastavení, pro každou požadovanou proměnnou se v této části nastavuje

její hodnota. Všechna specifická nastavení jsou hře předána při jejím spuštění.

Obrázek 9 Případy užití modulu GameControl



Poslední část obsahuje potřebné věci přímo pro výběr hry a její spuštění. Po vybrání hry ze seznamu všech dostupných her se v seznamu výstupních zařízení označí všechna hrou podporovaná výstupní zařízení. Lze z nich zvolit pouze podmnožinu, hra nemusí nutně běžet na všech podporovaných zařízeních, ale musí zůstat označené alespoň jedno. Pro každé z vybraných výstupních zařízení systém provede kontrolu jeho připravenosti a zobrazí informaci o jeho aktuálním stavu. Pokud jsou všechna vybraná výstupní zařízení připravena, je připojeno vstupní zařízení, jsou nastavena všechna specifická nastavení hry a bylo provedeno nastavení pásem, je možné spustit hru.

5.2.10 TabletControl

Modul slouží jako připojovací slot pro mobilní zařízení, prostředníctvím kterého pak může aplikace komunikovat s inteligentním robotem.

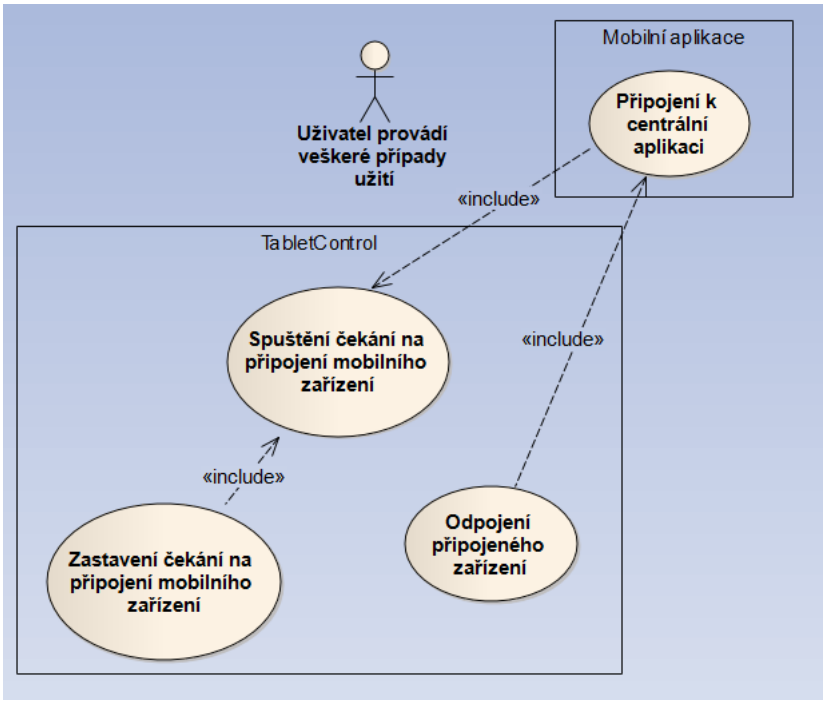
Ve výchozím stavu je modul neaktivní. Uživatel aplikace ho může přepnout do stavu, ve kterém čeká na připojení mobilního zařízení. Modul informuje uživatele o tom, na jaké adrese a portu připojení zařízení očekává. Na mobilním zařízení se tyto údaje zadají do příslušných vstupních polí a systém se pak spojí s mobilním zařízením.

Po úspěšném spárovnání s mobilním zařízením se modul přepne do stavu připojeno, zobrazí základní informace o spárovaném zařízení a informuje uživatele o stavu propojení mezi robotem a mobilním zařízením.

Teprve ve chvíli, kdy je v pořádku jak spojení mezi aplikací a mobilním zařízením, tak mezi mobilním zařízením a robotem, modul informuje ostatní moduly, že je připraven.

Pokud v modulu GameControl chceme sputit hru, jejímž výstupním zařízením je právě robot, bude stav tohoto modulu korespondovat se stavem, který zobrazuje GameControl u zařízení robota.

Obrázek 10 Případy užití modulu TabletControl



5.2.11 Moduly her

Jednotlivé hry jsou do aplikace přidávány jako součásti modulu GameControl. Nejedná se tedy přímo o moduly platformy NetBeans. Na druhou stranu jsou to samostatné jednotky, které s Hubem a ostaními potřebnými moduly komunikují přes veřejné rozhraní, které je shodné pro všechny hry.

Rozhraní zajišťuje, že hra bude schopná poskytnout ostatním modulům veškeré potřebné informace pro její nastavení a zároveň ji bude možné ovládat.

Spolu se zařazením do modulu GameControl je potřeba vložit o hře informace do databáze. GameControl modul pak na základě těchto informací může hru přes rozhraní spuštět a nahrávat tak jednotlivá kola sezení.

6 Implementace

V této kapitole se věnuji fyzické struktuře databáze a implementačním řešením jednotlivých modulů aplikace. Jsou stručně popsány způsoby, jakými jsou tvořeny jednotlivé obrazovky modulů a částečně jsou nastíněny pochody na pozadí a vzájemné interakce mezi moduly. U vybraných modulů byly vytvořeny diagramy tříd s jejich závislostmi.

6.1 Struktura databáze

Fyzický databázový model je implementací návrhového entitního modelu a jeho schéma je zobrazené na obrázku 11. Z jednotlivých entit se staly databázové tabulky, vznikly spojovací tabulky pro M:N vztahy a došlo k rozpadu varianty nastavení na tři různé tabulky.

Spojovací tabulka mezi klientem (Client) a diagnózou (Diagnose) mimo klíčů z obou spojovaných tabulek obsahuje ještě nepovinný atribut komentáře. Je tak možné klientovi k jeho diagnózám vkládat poznámky.

Stejným způsobem spojovací tabulka mezi kolem (Round) a nastavením kanálů (Channelsetting) obsahuje timestamp, který říká, kdy ke změně na dané nastavení kanálů došlo. Nastavení kanálů lze měnit i v průběhu kola a pro zpracování dat v rámci analýz po skončení nahrávání kola je potřebné vědět, jaké nastavení kanálů bylo použito pro výpočet hodnoty skóre v určitém čase kola.

K rozpadu entity varianty nastavení došlo právě z důvodu potřeby ukládat nastavení kanálů přímo ke kolům sezení a zároveň proto, že více variant nastavení může sdílet jak stejné nastavení kanálů, tak stejné nastavení pásem (Bandsetting). Databáze tak tedy je ve 3NF[1].

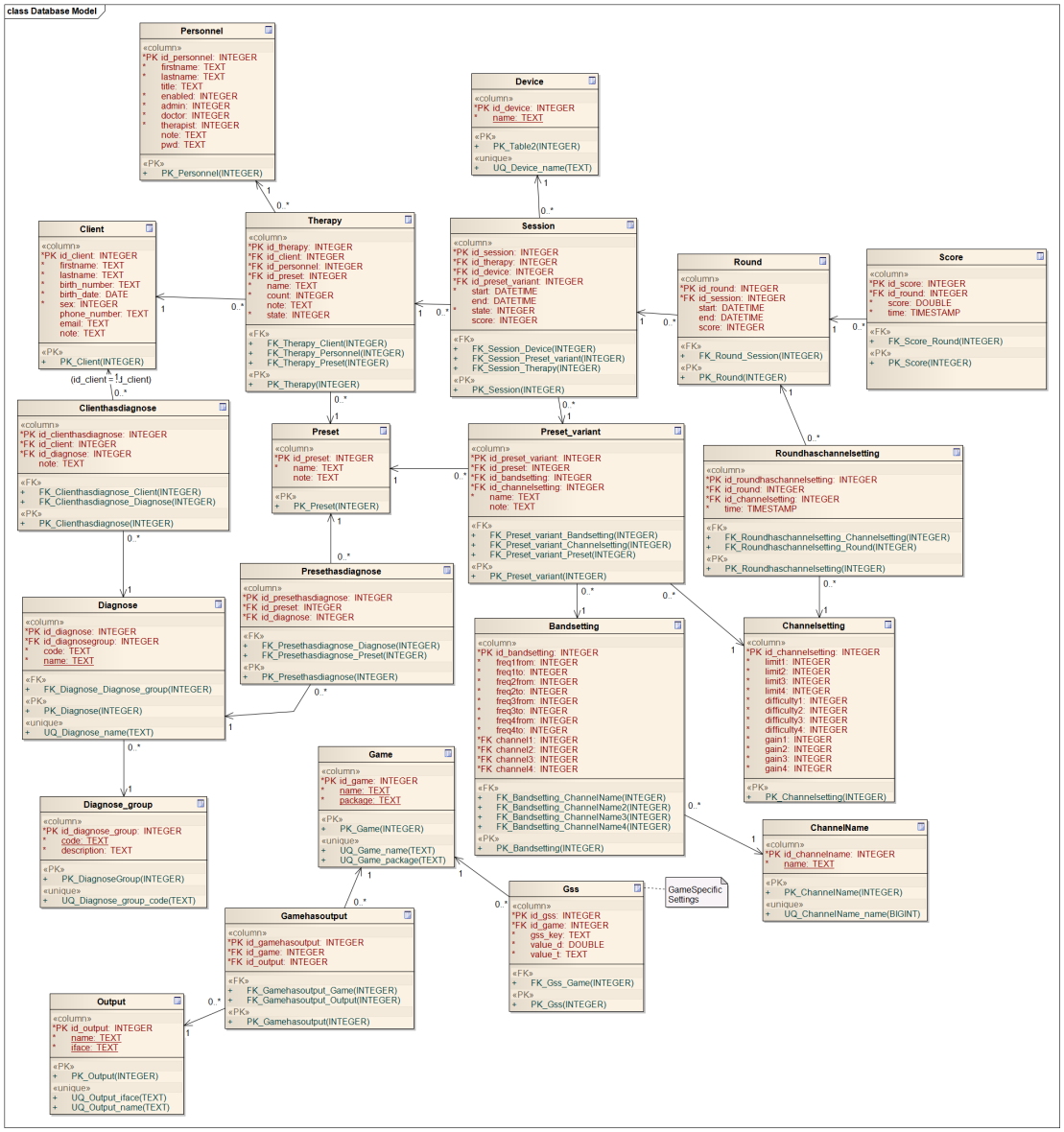
6.2 Moduly

NetBeans platforma vyžaduje, aby každý modul, který je součástí grafického uživatelského rozhraní, byl potomkem třídy TopComponent. Tato závislost zajistí, že platforma je schopná vytvořit okno modulu, a toto okno pozicovat spolu s ostatními moduly. Zároveň definuje několik základních vlastností každého modulu jako jsou například nadpis okna modulu, jeho výchozí pozice v rámci aplikace nebo zda-li je modul otevřen ihned po spuštění celé aplikce. Jediné dva moduly, které nemají svou TopComponentu jsou Bus a Hub.

Všechny moduly mimo Hub a Bus si během vytváření svého uživatelského rozhraní zároveň vytvoří i instanci třídy Communicator z Bus modulu. Protože je zaručeno, že každý modul bude mít vždy pouze jednu instanci třídy dědící z TopComponent, je tento komunikátor uložen jako statická proměnná právě v té třídě. Pro snadný přístup ke komunikátoru je pak v každém modulu veřejná metoda vracející objekt komunikátoru. Kterákoliv součást modulu tedy může přes komunikátor kdykoli odesílat zprávy.

[1] Třetí normální forma

Obrázek 11 Fyzický model databáze



6.2.1 Komunikační protokol a Bus modul

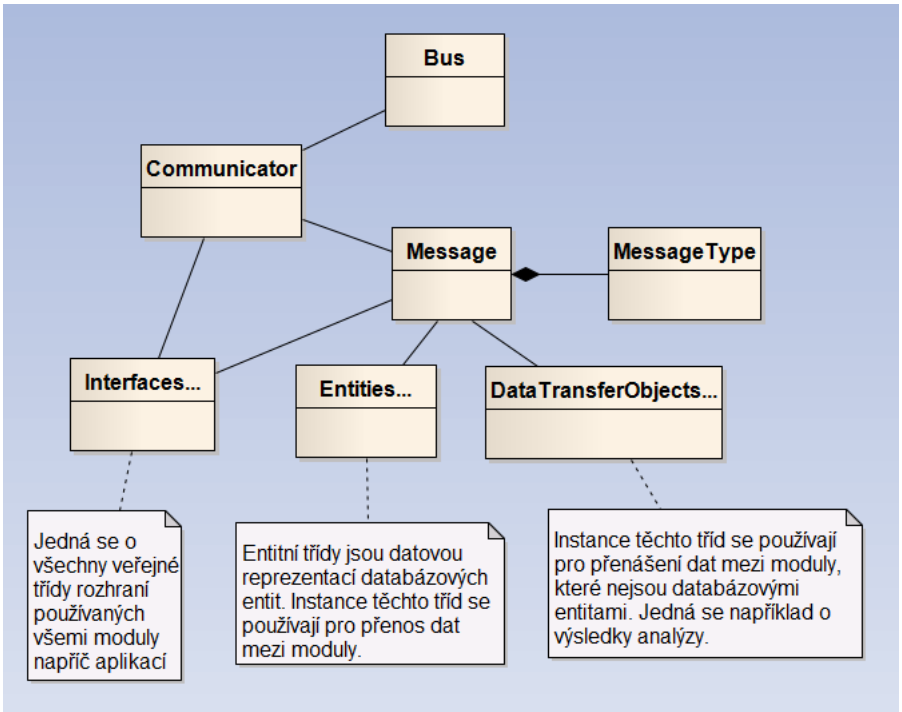
Modul, na kterém jsou závislé všechny ostatní moduly v aplikaci. Jeho nejdůležitější součástí jsou sdílené balíčky tříd rozhraní, entitních tříd a DTO[2] tříd.

Třída sběrnice (Bus) obaluje lookup platformy NetBeans a vytváří sdílený kanál sama se sebou, do kterého umí odesílat zprávy. Zároveň umí přidávat odběratele zpráv. Při přijetí zprávy informuje všechny přihlášené odběratele, že dorazila nová zpráva a je na každém odběrateli, aby zprávu zpracoval.

Komunikátor se při vzniku každé svojí instance přihlásí na sběrnici jako odběratel zpráv. Je na každém modulu, aby si v komunikátoru zaregistroval objekty, které budou zpracovávat zprávy zaslané na určité rozhraní. Komunikátor sám se pouze postará o to, aby při přijetí zpráv byly volány příslušné metody adresovaných rozhraní. Co se v reakci na volání funkce bude dít je čistě v režii modulu.

[2]DTO – Data Transfer Object – jednoduché objekty pro přenos informací

Obrázek 12 Class diagram pro Bus modul



Objekt zprávy povině vyžaduje pouze dva parametry – typ zprávy a rozhraní, pro které je zpráva určena. Dále záleží na typu zprávy, jaké další parametry zpráva musí mít. Pro každý typ zprávy má třída Message vhodný konstuktor. Díky návrhu jednotlivých rozhraní lze kód používající třídu Message a třídy rozhraní psát kompletně pomocí referencí, tudíž refaktorovatelně. Každé rozhraní má v sobě Enum se jmény svých metod. To je potřeba například při vytváření objektu Message s typem CONTROL. Takové zprávě je třeba jako parametry dát jméno cílového rozhraní, jméno metody onoho rozhraní, která se má zavolat, a parametry pro její volání. Díky enumům v třídách rozhraní lze jméno metody zadávat jako odkaz na prvek enumu. V případě, že by bylo potřeba metodu v rozhraní přejmenovat, přejmenuje se zároveň s ní i prvek enumu a spolu s ním se zrefaktorují i všechna jeho použití při vytváření zpráv.

6.2.2 UserControl

Jak bylo zmíněno v analýze, jedná se o velmi jednoduchý modul, který umožňuje jednotlivým uživatelům přihlašování a odhlašování se do a ze systému.

Teoreticky by v aktuálním stavu mohl klidně tento modul být součástí modulu ClientControl. Důvodů k vytvoření samostatného logovacího modulu bylo několik. Z hlediska přidávání dalších modulů, které budou vyžadovat přihlášení uživatele, je žádoucí, aby bylo přihlašování nějak centralizované. Dalším pozitivem je, že jako samostatný modul může být přihlašovací formulář libovolně přemísťován, uživatel si ho napozicuje dle svých zvyklostí a navíc může být drtivou většinu času schovaný v jedné z postraních lišt. Kdykoli ho bude za potřebí, lze ho okamžitě vyvolat stisknutím příslušného tlačítka na liště.

Modul sám o sobě je vlastně jedním panelem s roletkou, do které jsou nahrány položky databázové tabulky Personnel, TextFieldem pro zadání hesla, tlačítkem pro přihlášení a odhlášení a dvěma informačními oblastmi.

Po vyplnění údajů a stisknutí tlačítka modul odešle po protokolu zprávu na Hub, která říká, že se má zalogovat uživatel s předanými údaji. Hub ověří správnost zadaného hesla vůči databázi. Pokud data neodpovídají, pošle zprávu UserControlu, že zadané heslo neodpovídá, nelze tedy uživatele přihlásit. Je-li heslo zadáno správně, pošle zprávu na rozhraní IUserLoginNotiff, čímž informuje všechny moduly o tom, že uživatel byl úspěšně přihlášen. Pro potřeby modulů je pak ve zprávě přiložen entitní objekt uživatele.

6.2.3 ClientControl

Modul správy klientů, jejich diagnóz, terapií, sezení a kol. Jak naznačuje class diagram na obrázku 13, jedná se o nejsložitější modul celé aplikace. Důvodem je velký počet případů užití (přehled na diagramu případů užití, obrázek 6) a s tím související množství dat, které je nutné zobrazovat a dále s nimi pak pracovat. Cílem návrhu modulu bylo, aby všechny možné operace byly na sobě pokud možno nezávislé a šlo je provádět v libovolném pořadí popřípadě zároveň. Je také potřeba, aby si modul pamatoval stavy jednotlivých sekcí, aby byly zobrazeny v nezměněném stavu, pokud si je uživatel vyžádá v libovolném okamžiku. Toho je dosaženo díky existenci různých panelů pro různé stavy jednotlivých částí každé sekce modulu a díky důmyslnému systému jejich vrstvení a vyměňování.

Dále jsem se snažil minimalizovat vytěžování intermodulárního protokolu a veškeré požadavky na databázi se provádí až ve chvíli, kdy jsou nezbytně nutné.

Jak bylo zmíněno v analýze modulu, skládá se ze společné části (časové osy) a tří sekcí – správy klientů, správy sezení a správy kol.

Časová osa je vždy zobrazena v horní části modulu a slouží pro navigaci mezi jednotlivými sekcemi modulu a jako rychlý ukazatel stavu celého modulu. Skládá se ze tří tlačítek typu JButton, které ale mění svou vizuální podobu podle aktuálního stavu komponenty a celého modulu. Stavy jsou jako v ostatních případech řešeny pomocí implementace zjednodušené verze návrhového vzoru State Pattern [19].

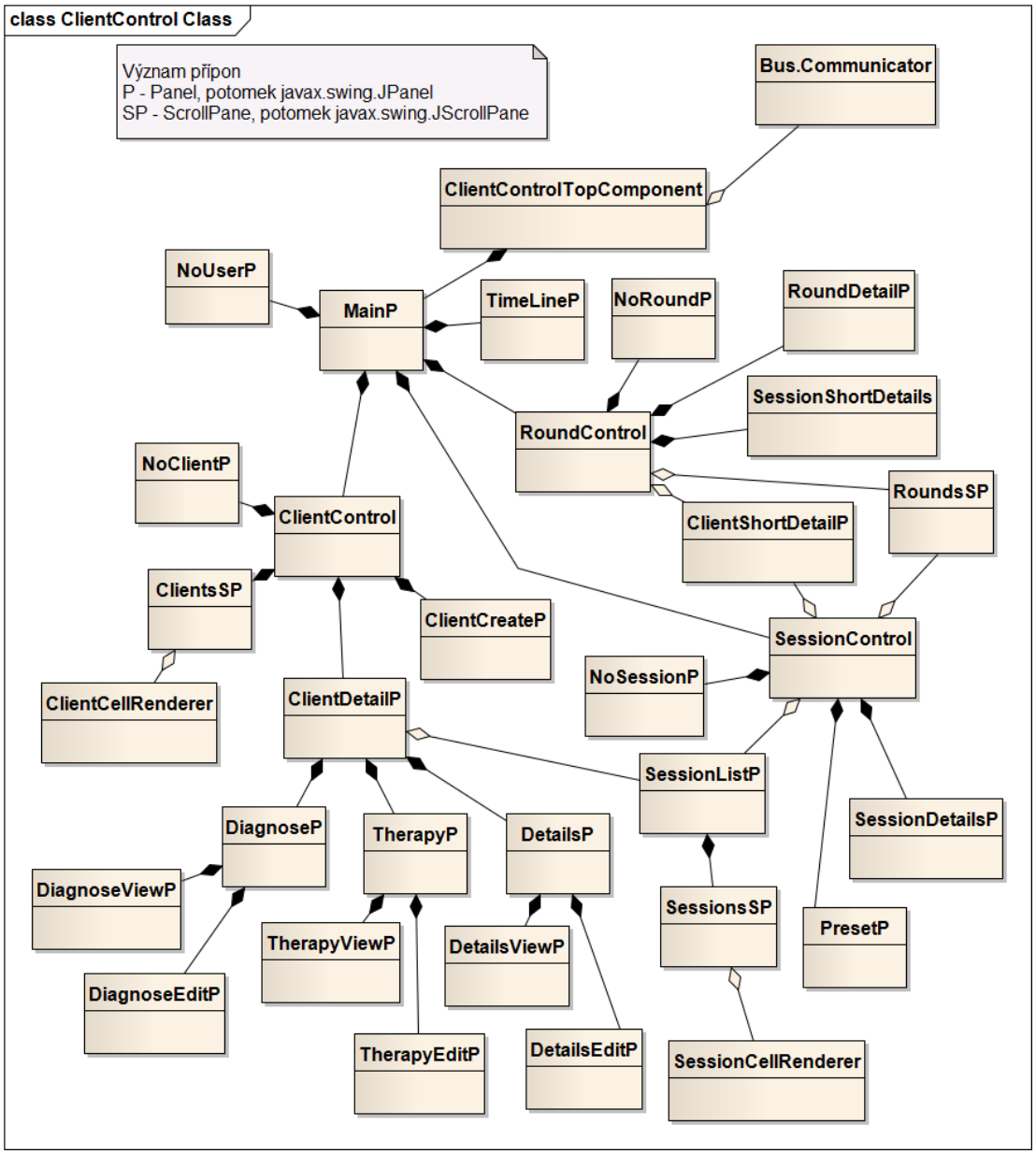
Dokud v rámci modulu není vybrán žádný klient, je na časové ose povoleno a vybarveno pouze tlačítko klientů, ostatní jsou zakázána a se šedým pozadím. Jakmile dojde k vybrání klienta, povolí se tlačítko sezení, stejně tak v okamžiku, kdy je v rámci modulu vybráno některé sezení, povolí se tlačítko kol. Při povolení se tlačítka podbarví modrou barvou a tím umocňují vizuální stránku zobrazení stavu modulu. Při změně výběru klienta dojde ke zrušení výběru sezení (bylo-li nějaké vybrané) a proto při něm dojde k zakázání tlačítka kol.

Stisknutím některého z tlačítek se převede časová osa do korespondujícího stavu a modul zobrazí odpovídající sekci. Právě aktivní tlačítko na časové ose vizuálně působí jako neustále zmáčknuté a je podbarvené zeleně (zeleně jsou v aplikaci zobrazovány všechny uživatelské výběry). Celá časová osa tedy svým vzhledem napomáhá rychlé orientaci po modulu a de facto působí jako simulace záložkového rozložení.

Základní rozložení sekce správy klientů je zajištěno pomocí čtyř objektů typu JSplitPane. První odděluje zobrazení listu klientů od zobrazení detailů vybraného klienta, druhá rozděluje oblast detailů na dvě nad sebou ležící oblasti, v každé z nich je pak další JSplitPane. V horní části je vlevo oblast věnovaná detailům klienta, vpravo pak seznam sezení vybraného klienta. Ve spodní části vlevo jsou diagnózy klienta a vpravo terapie klienta. Právě díky umístění na JSplitPane lze tažením myší přizpůsobovat velikost jednotlivých částí podle aktuálních potřeb přímo za běhu aplikace.

Seznam klientů je potomkem třídy JScrollPane a zachovává tak její funkčnost co se týče zobrazování do ní vložených komponent. Do ní je umístěn JList, pro jehož vykreslování je implementována speciální verze třídy ListCellRenderer. Ta definuje, jak

Obrázek 13 Class diagram pro modul ClientControl



se mají zobrazovat jednotlivé položky listu. V listu tedy u každého klienta je vidět jeho celé jméno, rodné číslo, počet jeho terapií a počet jeho sezení. Nad listem klientů je JTextField umožňující rychlé vyhledávání. Vyhledávat lze podle křestního jména, příjmení, jejich kombinací nebo podle rodného čísla. List reaguje na veškeré změny ve vyhledávacím políčku okamžitě a zobrazuje v seznamu pouze klienty, kteří odpovídají vyhledávané frázi. Pod seznamem je pak SwitchBox, jehož přepínáním si uživatel vybírá, zda-li chce zobrazit pouze své klienty (tj. ty, kteří s ním mají alespoň jednu terapii) nebo všechny dostupné.

Detaily klienta mají dvě podoby – zobrazovací a editační. V zobrazovacím módu jsou na panel detailů vygenerovány objekty typu JLabel s informacemi o klientovi uloženými v databázi. Pomocí tlačítka Editovat lze detaily přepnout do editačního módu. Panel s JLabely je nahrazen totožně vypadajícím panel, který ale místo JLabelů má data zobrazená v objektech typu JTextField, takže je uživatel může libovolně upravovat. Namísto

tlačítka Editovat jsou dvě další, jedno pro potvrzení změn a druhé pro navrácení zpět do zobrazovacího módu bez uložení změn.

Oblast diagnóz má stejně jako detaily zobrazovací a editační mód. V zobrazovacím módu je pro každou z diagnóz přiřazených klientovi zobrazen jeden řádek obsahující kód diagnózy (její diagnostické skupiny), její jméno, vloženou poznámku a dvě tlačítka. První umožňuje upravovat korespondující diagnózu, druhé jí úplně vymazat ze systému. Pod záznamy diagnóz je tlačítko pro přidání nové diagnózy. Stisknutím editačního tlačítka jedné z diagnóz nebo tlačítka přidání diagnózy se oblast přepne do editačního režimu. Je zobrazeno textové políčko pro poznámku terapeuta a dvě roletkové nabídky, jedna pro skupinu (kód) diagnózy, druhá přímo pro diagnózu. Obsah roletky s diagnózami se mění podle hodnoty vybrané v roletce diagnostických skupin a urychluje tak hledání správné diagnózy. Při editaci diagnózy jsou zobrazeny hodnoty odpovídající jejím aktuálním parametrům. Na panelu jsou i dvě tlačítka, jedno pro potvrzení změn nebo vložení nové diagnózy a jedno pro návrat zpět do režimu zobrazení diagnóz.

Oblast terapií je funkčností víceméně totožná s oblastí diagnóz. Rozdílem je, že terapie nelze ze seznamu mazat, lze je pouze přidávat a upravovat. V editačním módu, který se zobrazí prázdný po stisknutí tlačítka přidat terapii nebo s předvyplněnými hodnotami po stisknutí tlačítka editace u některé z existujících terapií, je formulář, který umožňuje nastavovat všechny parametry terapie. V roletkových nabídkách se vybírá terapeut, stav terapie a přednastavení parametrů terapie. V roletce nastavení se zobrazují pouze položky, které korespondují alespoň s jednou diagnózou pacienta. Založit terapii pro pacienta, který dosud nebyl diagnostikován proto není možné. Pro jméno terapie a poznámky k terapii jsou k dispozici textová políčka. Pro zadávání počtu sezení je přítomen objekt typu JSpinner, do kterého lze hodnotu buď napsat ručně nebo naklikat pomocí šipek v jeho pravé části. Stejně jako u ostatních částí sekce i zde jsou dvě tlačítka – jedno pro potvrzení změn a druhé pro návrat do módu zobrazení terapií.

Poslední oblastí sekce správy klientů je seznam sezení. Stejně jako seznam klientů je seznam sezení potomkem třídy JScrollPane, na kterou je umístěn JList. Pomocí další varianty implementace třídy ListCellRenderer bylo u každé položky seznamu dosaženo zobrazování několika základních informací o příslušném kole. V seznamu jsou všechna sezení pacienta ve všech terapiích seřazená chronologicky. Seznam lze přepnout na zobrazení sezení pouze jedné terapie a to za pomoci výběru z roletkové nabídky přímo pod seznamem sezení. V té jsou k dispozici všechny existující terapie klienta a navíc položka pro zobrazení všech sezení bez ohledu na terapii. Každá změna ve filtrovací roletce se okamžitě potvrzuje. Při vybrání sezení ze seznamu se zpřístupní sekce správy kol modulu. Dvojklikem na položku sezení v seznamu je možné přejít k detailům sezení v sekci správa sezení.

Dokud není v seznamu klientů vybrána nějaká položka, na místě detailů klienta je vložen panel informující uživatele, že pro práci se sekcí je potřeba nějakého klienta vybrat nebo je potřeba založit nového. Vyvolat formulář pro založení nového klienta lze buď tlačítkem na zmíněném panelu nebo stejnojmenným tlačítkem pod seznamem klientů.

Vytvářecí formulář je v zásadě stejný jako editační formulář informací klienta, jen je zobrazen přes celou oblast detailů klienta – během vytváření klienta nemá smysl zobrazovat oblasti pro diagnózy, terapie ani seznam sezení. Při potvrzení vytvoření klienta příslušným tlačítkem je tento nově vytvořený klient automaticky vybrán jako aktivní a jsou zobrazeny jeho detaily.

Sekce správy sezení je v mnohém podobná sekci správy klientů. To byl cíl při jejím návrhu, práce v obou sekcích je tak téměř stejná a uživatelům aplikace trvá kratší dobu přivyknout na způsob, jakým je s daty zacházeno. Sekce správy sezení je rozložena

pouze na třech objektech typu JSplitPane a obsahuje tak o jednu oblast méně, než sekce klientů.

První JSplitPane rozděluje prostor sekce vertikálně, v levé části je seznam sezení, v pravé pak detaily sezení. Seznam sezení je totožný jak zobrazením tak funkčností se seznamem sezení v pravé horní části sekce správy klientů. Nad seznamem sezení je malý informační panel s daty vybraného klienta. Dvojklikem na tento panel lze přejít na zobrazení detailů klienta. Druhá JSplitPane dělí oblast detailů sezení znovu vertikálně. V pravé části je seznam kol sezení, v levé poslední JSplitPane, která dělí zbylý prostor horizontálně, na oblast zobrazení informací o samotném sezení a na oblast varianty nastavení terapie.

Seznam kol je po vzoru ostatních seznamů potomkem JScrollPane a má svou vlastní implementaci třídy ListCellRenderer. Má tedy zase unikátní zobrazování jednotlivých položek seznamu tak, aby se zobrazovaly relevantní informace o kolech. Dvojklikem na položku v seznamu lze přejít na detail vybraného kola do sekce správy kol.

Oblast informací o sezení je jedinou zobrazovací oblastí, která nemá editační mód. Jediným parametrem, který uživatel může u již vytvořeného sezení měnit totiž je jeho stav. Změna stavu se provádí výběrem nového stavu z roletkové nabídky a stisknutím tlačítka Uložit. Možné stavy sezení jsou tři. Po vytvoření je sezení ve stavu „Probíhající“, jeho začátek je okamžik vytvoření sezení a jeho konec není definován. Při přepnutí do stavu „Ukončené“ nebo „Stornované“ se nastaví konec kola na okamžik, kdy byl změněn jeho stav. Zároveň se spočítá celkové skóre sezení na základě jeho proběhnutých kol a hodnot jejich skóre.

Každé sezení je asociováno s jednou variantou nastavení pásem a kanálů. V poslední oblasti sekce správy kol se zobrazují informace o aktuálně nastavené variantě. Varianta nastavení se skládá ze dvou částí – nastavení pásem a nastavení kanálů – a pro každou je zobrazován vlastní panel, na kterém jsou přehledně vypsané hodnoty všech vlastností. Oblast lze přepnout do editačního módu stisknutím tlačítka Změnit. V editačním módu se jméno terapie změní na roletkovou nabídku obsahující veškeré varianty nastavení parametrů pro vybrané přednastavení parametrů terapie, do které patří zobrazené sezení. Při výběru některé varianty z roletky se na příslušných panelech v oblasti zobrazí detaily vybrané varianty. Změnu varianty je třeba potvrdit tlačítkem Uložit, pomocí tlačítka Zrušit se lze vrátit do zobrazovacího módu aniž by se provedené změny promítly do objektu sezení.

Stejně jako u klientů, dokud není vybráno některé ze sezení, na místě detailů sezení se zobrazuje pouze informační panel. Na něm je umístěno tlačítko pro vytvoření nového sezení. Po jeho stisknutí, stejně jako po stisknutí tlačítka pod seznamem sezení, se na místě detailů sezení zobrazí formulář pro vytvoření nového sezení. Při zakládání nového sezení je potřeba vybrat z roletkového menu terapii, do které bude nové sezení patřit. Na základě výběru terapie jsou v následující roletce zobrazeny jednotlivé varianty nastavení parametrů v závislosti na vybraném přednastavení parametrů vybrazné terapie. V poslední roletce jsou zobrazena dostupná zařízení v systému. Jedná se tady pouze o administrativní popisek vstupních zařízení, tato nabídka není nijak spojena s nabídkou vstupních zařízení v modulu DeviceSettings. Po potvrzení vytvoření nového sezení je sezení automaticky vybráno a zobrazí se jeho detaily. Vybrané zařízení ani terapie, do které sezení patří, už nelze měnit.

Sekce správy kol je poslední sekcí modulu a je ze všech sekcí nejjednodušší. Celá se sestává pouze ze dvou částí umístěných na jediné JSplitPane. V levé části je seznam kol, opět shodný se seznamem kol v sekci správy sezení. Nad ním je znovu malý informační panel s informacemi o klientovi se zachovanou funkčností ze sekce správy sezení. Pod ním je ale ještě další panel informační panel, na kterém jsou informace o aktuálně vybraném

sezení. Dvojklikem na něj lze přejít na zobrazení detailů sezení v sekci správy kol.

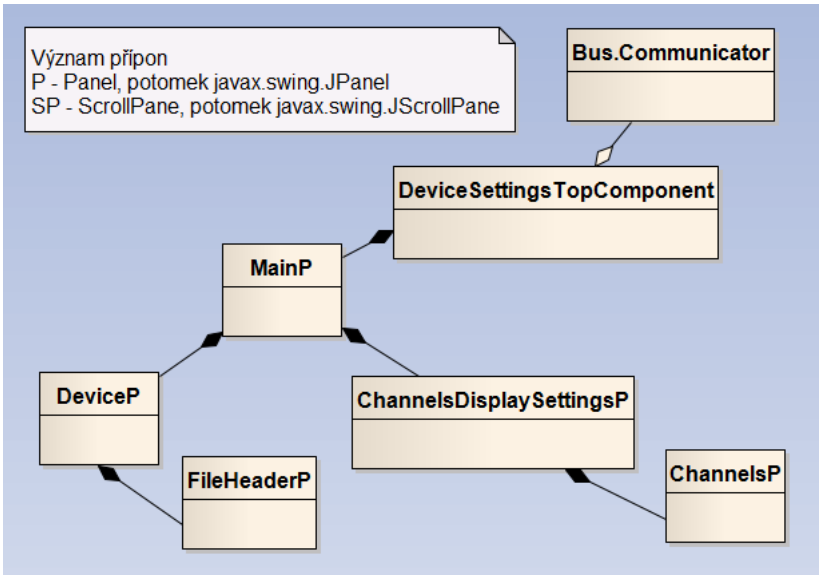
V pravé části sekce je prostor pro detaily kola. Zobrazení se liší podle stavu kola. Pokud kolo ještě nebylo nahráno a nejsou k němu výsledky, je na místě detailů jen panel který odkáže uživatele k nahrání kola přes modul GameControl. Pokud je kolo ukončeno, je vykreslen graf jeho průběhu a uživateli nabídnuta možnost vytvořit a spustit další kolo.

Vytvoření nového kola nemá žádný speciální formulář. Po stisknutí některého z tlačítek označených „Nové kolo“, dostupných pod seznamy kol jak v sekci správy kol tak v sekci správy sezení, na informačním panelu před vybráním kola ze seznamu nebo pod grafem ukončeného sezení, se ihned vytvoří nové kolo, automaticky se označí jako aktivní a zobrazí se jeho detaily.

6.2.4 DeviceSettings

Další z jednodušších modulů, ale jeho samostatná existence je stejně jako u UserControlu opodstatněná. Přes tento modul se centrálně připojují vstupní zařízení. Modulů používajících vstupní zařízení bude výhledově celá řada.

Obrázek 14 Class diagram pro modul DeviceSettings



Modul má dvě části. Jednou je ovládací panel připojování zařízení a druhou panel pro výběr kanálů, které se mají zobrazovat. Dokud není připojeno nějaké vstupní zařízení, není panel výběru kanálů vidět.

Panel na připojování zařízení umožňuje připojit zařízení přímo do systému přes modul Hub, nebo načíst nahraná data ze souboru. Výběr mezi těmito možnostmi je zajištěn dvěma RadioButtony, které na základě toho, který z nich je vybrán, aktivují a deaktivují ostatní vstupní prvky na panelu.

Pokud je vybráno přímé připojení zařízení, stačí vybrat z roletky jedno z dostupných zařízení a pak ho stisknutím příslušného tlačítka připojit. Je-li zvoleno otevření souboru, aktivuje se pole pro zadání cesty k souboru. Samozřejmě lze použít grafické vyhledávání souboru pomocí instance třídy JFileChooser. Po připojení souboru se na panel hlavičky souboru vypíšou informace o zařízení nalezeném v souboru.

Stisknutím tlačítka „Potvrdit výběr zařízení“ se pošle zpráva na modul Hub, který vybrané zařízení připojí. Zároveň pošle zprávu na rozhraní IDeviceChange, čímž in-

formuje všechny moduly o připojení vstupního zařízení. Ve zprávě je přiložen entitní objekt připojeného zařízení.

Připojením zařízení se povolí tlačítko pro spuštění signálů a zobrazí se panel pro výběr zobrazovaných kanálů. Jeho obsah je automaticky generován z informací připojeného vstupního zařízení, pro každý kanál zařízení je zobrazeno zaškrtávací políčko se jménem kanálu. Ve výchozím stavu jsou zobrazovány všechny kanály zařízení. Změny na panelu se promítají okamžitě, každá změna stavu pošle zprávu ostatním modulům s polem hodnot typu boolean určujícím, které kanály mají být vykreslovány. Ve chvíli, uživatel zmáčkne tlačítko „Spustit signál“, modul pošle zprávu na Hub, který pak začne přijímat a zpracovávat data z vybraného vstupního zařízení a odesílat zprávy typu „DATA“ s objekty RawDTO a s objekty QualityDTO. RawDTO obsahují přímo přijatá data ze zařízení pro všechny kanály, QualityDTO pak informaci o kvalitě příjmu signálu z elektrody pro každý kanál.

Se spuštěním zpracování signálů se odemkne možnost spuštění analýzy. Stisknutím tlačítka „Spustit spektra“ se odešle zpráva na Hub, který pak začne protékající data analyticky zpracovávat a výsledky analýzy odesílat ostatním modulům ve formě datových zpráv s objekty typu DrawSpectrumDTO.

6.2.5 SignalDisplay

Modul pro zobrazování dat ze vstupního zařízení. Obsahuje zjednodušenou variantu návrhového vzoru State Pattern [19]. Modul má dva stavy, „NO_DEVICE“ pokud není připojeno žádné vstupní zařízení a „READY“, pokud zařízení bylo připojeno.

Obsah jeho modulového okna (potomek TopComponent) se mění v závislosti na stavu modulu. Nemá-li modul zařízení, je zobrazen pouze panel s informativní zprávou, jinak pak panel s grafy, do kterých se kreslí příchozí signály. Panel s grafy má díky umístění na panel typu ResizablePanel obohacenou funkčnost a lze ho libovolně zvětšovat a zmenšovat tažením za jeho pravý spodní roh. Jeho obsah se automaticky přizpůsobuje jeho aktuální velikosti.

Samotný panel grafů je pak konstrukčně složitější i když jeho výsledná podoba působí velmi jednoduše. Složitost je způsobena knihovnou, přes kterou se grafy kreslí a její neschopnost dosáhnout požadovaného výsledku konvenční cestou nastavování vlastností grafů. Pro každý kanál vstupního zařízení, který je vybrán k zobrazování, se zobrazuje jedna linka. Je potřeba mít všechny linky co nejblíže u sebe, zarovnané pod sebe tak, aby stejné pozice na vodorovné ose byly u všech grafů shodně nad sebou a aby se jednotlivé čáry nepřekrývaly. Místo kreslení všech čar do jednoho grafu jsem proto zvolil cestu více grafů a pro každý kanál zařízení je na panelu vlastní graf. Problémem s umístěním jednotlivých grafů je, že i v úplně minimalistické verzi knihovna vykresluje graf s poměrně velkými okraji a ve výsledku tak mezi jednotlivými grafy byly mezery, které nejenže zbytečně zabíraly místo na obrazovce, takže grafy jako takové byly menší, ale navíc zhoršovaly čitelnost údajů.

Panel grafů je tedy poskládán tak, že na něm jsou nejprve popisky jednotlivých kanálů v podobě objektů třídy JLabel, za každým popiskem je JScrollPane, která je zbavená veškerých dekorací a má zakázáno zobrazování scrollbarů. Tvoří tak tedy pouze vykreslovací oblast, do které je pak vložen graf. Graf je sám potomkem třídy JPanel, takže s ním lze nakládat jako s klasickým panelem. Výsledného efektu je dosaženo tak, že jednotlivé grafy jsou větší, než jsou vykreslovací oblasti jejich JScrollPane. Protože je umístění na JScrollPane centrované, přebytečné části grafů, které by jinak na obrazovce překážely, jsou schované mimo vykreslovací oblasti. Výsledkem tedy je řada pod sebou umístěných grafů, které splňují veškeré původní pomínky.

Při změně velikosti panelu grafů se změna propaguje směrem dolů přes JScrollPane až k panelům jednotlivých grafů a velikost všech komponent je upravena tak, aby výsledek byl co nejlepší.

Měnit zobrazované kanály lze i přímo za běhu vykreslování signálů. Modul dostává data ze všech kanálů a kreslí grafy pro všechny kanály. Pokud tedy uživatel zapne zobrazení některého z kanálů, který byl do té doby vypnutý, nezačne se graf vykreslovat od okamžiku zapnutí, ale zobrazí se data stejně jako na ostatních kanálech, které byly zapnuté celou dobu.

6.2.6 FourierDisplay

Modul pro zobrazování výsledků analýzy dat ze vstupního zařízení. Stejně jako SignalDisplay, obsahuje jednoduchý State Pattern [19].

Modul potřebuje pro své fungování nastené parametry analýzy, a proto dokud nemá nastavení k dispozici je ve stavu „NO_SETTINGS“ a zobrazuje pouze informační panel, že není k dispozici nastavení spektra. v okamžiku, kdy dostane od modulu FourierSettings nastavení analýzy, přepne se do stavu „READY“ a zobrazí grafy pro výsledky analýzy.

Základní rozdíl oproti zobrazování nezpracovaných signálů v modulu SignalDisplay je v tom, že grafy výsledků analýzy jsou značně složitější a jednotlivé grafy jsou vykreslovány samostatně namísto v jednom bloku.

Stejně jako v modulu SignalDisplay je pro každý kanál vstupního zařízení vykreslován jeden graf. I u těchto grafů je možné zobrazení některých kanálů vypnout. Ovládání zobrazování kanálů je součástí modulu FourierSettings, aby ovládací panel nezabíral místo potřebné k vykreslování grafů. Ke grafům kanálů je navíc přidán jeden další, který je jejich souhrnem. Zobrazení souhrnového grafu nelze vypnout.

Modul je rozdělen do dvou částí pomocí JSplitPane, která umožňuje měnit velikost svých částí uživateli tažením za předělovací čáru. V horní části je pouze souhrnový graf, ve spodní části pak JScrollPane, na které je vykreslen panel s grafy. Grafy jsou na panelu umístěné ve třech sloupcích a pokud se v nějakém směru nevejdou na obrazovku, díky JScrollPane se zobrazí scrollbary a lze pohybovat zobrazovací oblastí.

Každý z grafů je umístěn na panelu typu ResizablePanel, lze tedy tažením za jeho pravý spodní roh měnit jeho velikost. Pokud jsou v rámci terapie některé kanály důležitější, lze je zvětšit aniž by bylo potřeba vypínat zobrazení ostatních kanálů.

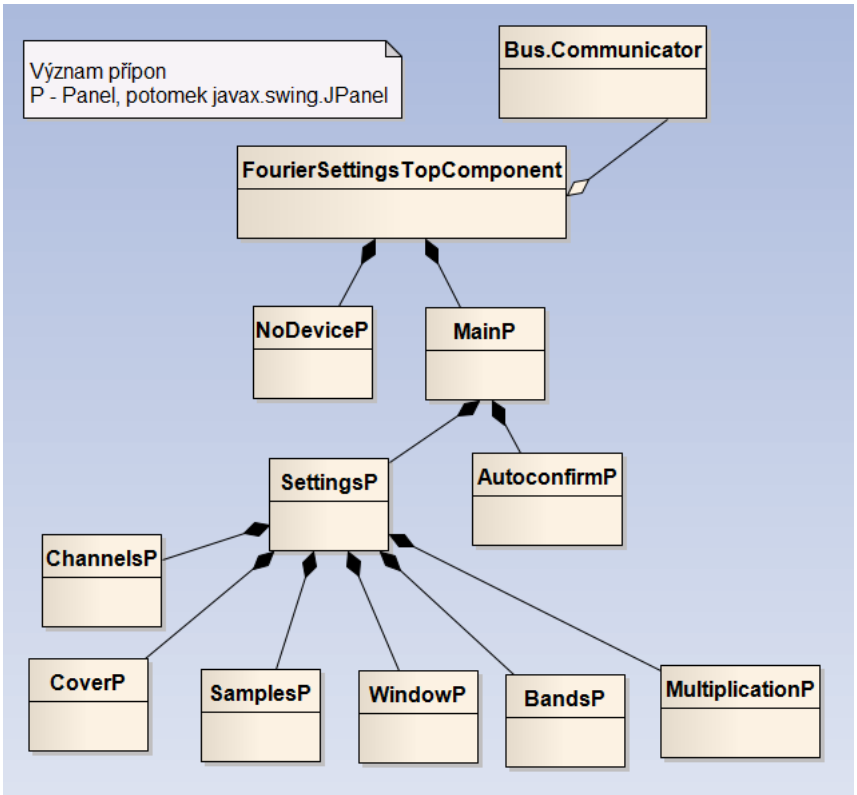
6.2.7 FourierSettings

Modul slouží pro nastavování parametrů analýzy dat ze vstupního zařízení. Pro své fungování potřebuje připojené vstupní zařízení a pro ovládání zobrazení stejně jako ostatní moduly používá zjednodušené varianty vzoru State Pattern [19]. Do připojení zařízení je ve stavu „NO_DEVICE“ a zobrazuje pouze informační panel, po připojení přejde do stavu „READY“ a zobrazí ovládací prvky nastavování parametrů analýzy.

Pro nastavení, které kanály se mají zobrazovat je použita stejná komponenta jako při výběru kanálů pro modul SignalDisplay. Pro každý panel se vygeneruje JCheckbox s popiskem, jejich zaškrtáváním a odškrtáváním se pak volí zobrazované kanály. Ve výchozím stavu jsou všechny kanály vybrané.

Nastavování pokrytí je možné dělat dvojím způsobem. Pro rychlé a pohodlné upravování hodnot slouží JSlider, pokud je potřeba zadat přesnou hodnotu, může být vyplněna do políčka vedle. Obě komponenty jsou mezi sebou obousměrně svázané, při změně jedné se upraví hodnota druhé.

Obrázek 15 Class diagram pro modul FourierSettings



Pro nastavení počtu vzorků je přítomen JSpinner. Pohybuje se po mocninách dvojky, ale lze do něj i přímo vepsat požadovanou hodnotu.

V sekci nastavování pásem je pro každé pásmo jeden posuvník. Jejich fungování je ale maličko složitější. První posuvník nastavuje spodní frekvenci pásma Delta. Druhý posuvník pak spodní hranici pásma Theta a zároveň horní hranici pásma Delta. Pro následující pásma platí to samé – posuvník nastavuje spodní hranici svého pásma a horní hranici předchozího. Horní hranice pásma Gamma se nastavit nedá, je pevně stanovená na 40 Hz.

Další částí je nastavení multiplikace. Multiplikace ve výchozím nastavení není povolena, nastavení jejích parametrů je proto neaktivní. Po zapnutí multiplikace přepnutím příslušného SwitchBoxu se zpřístupní pole parametrů multiplikace.

Poslední částí modulu je oblast potvrzování změn parametrů. Obsahuje jednak tlačítko, po jehož stisknutí se odešle zpráva informující ostatní moduly o změně nastavení parametrů analýzy, tak SwitchBox, který po přepnutí zapne automatické odesílání změn v nastavení. Veškeré komponenty nastavující parametry jsou kaskádně spojené systémem Listenerů a při automatickém odesílání změn se zpráva ostatním modulům odešle pokaždé, když uživatel dokončí změnu některého z parametrů.

6.2.8 GameControl

Modul spouštění her a nahrávání jednotlivých kol sezení. Jeho funkcionalita je poněkud složitější a to se odráží i do jeho struktury. Rozhraní modulu je opět pomocí objektů JSplitPane rozděleno do čtyř částí s nastavitelnou velikostí.

První částí je oblast výběru her. Je zde seznam všech dostupných her v aplikaci, panel pro výběr výstupních zařízení, panel předstartovní kontroly a ovládací tlačítka. Seznam

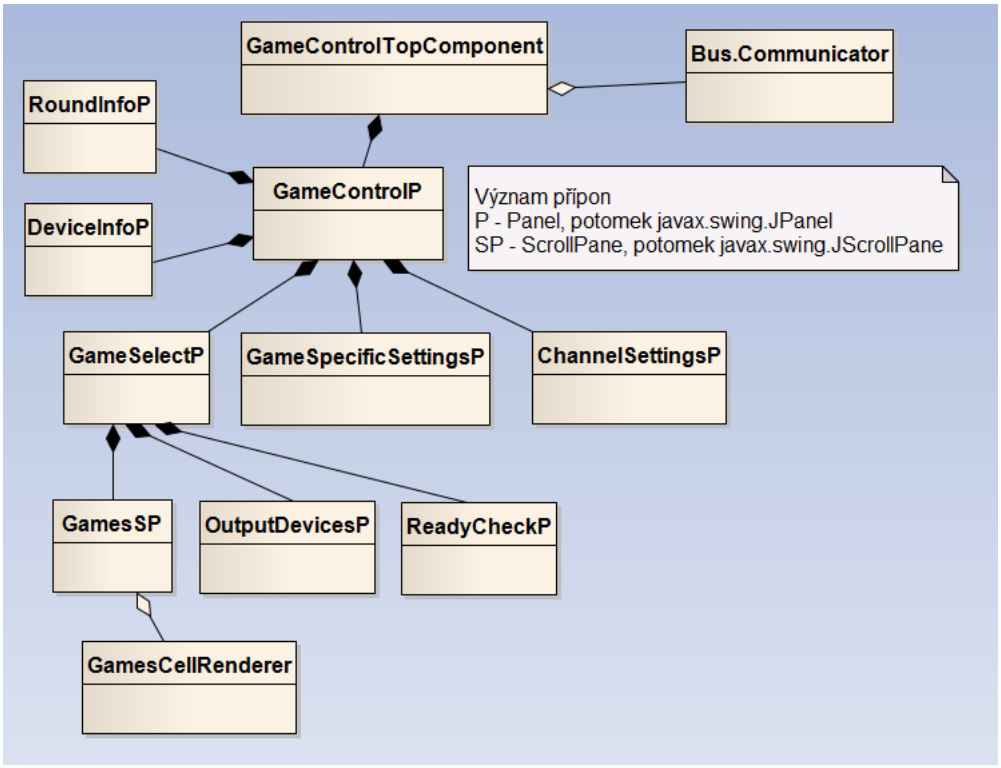
her je po vzoru ostatních seznamů implementován jako JList uložený na JScrollPane a pro vykreslování jednotlivých položek využívá dalšího typu ListCellRendereru tak, že pro každou hru zobrazí nejen její název ale i výstupní zařízení, která hra podporuje.

Po vybrání některé hry ze seznamu se v panelu výběru výstupních zařízení označí všechna zařízení, která hra podporuje a pro každé z nich se na panel předstartovní kontroly přidá jedna kontrolka. Dále je vybraná hra požádána, aby aplikaci poskytla panel svých nastavení, který se pak zobrazí v druhé části modulu a seznam kanálů, které hra využívá, podle kterého se povolují nebo zakazují skupiny komponent v nastavování parametrů kanálů. Uživatel může měnit výběr výstupních zařízení a podle toho se mění kontrolky na panelu předstartovní kontroly. Kontrolka vstupního zařízení přítomná u všech her, nehledě na vybraná výstupní zařízení.

Kontrolky na panelu předstartovní kontroly mají čtyři stavy.

- OFFLINE (šedivá barva) – zařízení je úplně odpojeno nebo vůbec nebylo nakonfigurováno
- WAITING (oranžová barva) – zařízení se připojuje nebo probíhá jeho konfigurace
- READY (zelená barva) – zařízení je připravené k provozu
- ERROR (červená barva) – na zařízení vznikla chyba

Obrázek 16 Class diagram pro modul GameControl



Vstupní zařízení a každý typ výstupního zařízení mají v aplikaci určena svá rozhraní a zasláním zprávy na příslušné rozhraní modul správy her zjišťuje stavy jednotlivých zařízení. Zároveň pokud dojde ke změně stavu zařízení, odešle zprávu na rozhraní GameControl se jménem svého rozhraní a novým stavem. Modul správy her je tak ihned o změně informován a upraví zobrazení kontrolek.

Tlačítko pro spuštění hry je implicitně zakázané. Pro spuštění hry je zapotřebí, aby bylo připojeno a aktivní vstupní zařízení a aby všechna vybraná výstupní zařízení byla připravena přijímat signály. K jeho povolení tedy dojde pouze ve chvíli, kdy všechny

kontrolky v oblasti předstartovní kontroly jsou ve stavy READY. Pro spuštění hry není nezbytně nutné, aby bylo vybrané kolo v rámci nějakého sezení, hra jde spustit i bez toho, ale její výsledky nebudou nahrávány a záznam nebude uložen do databáze.

Jak jsem již zmínil, druhou částí modulu je oblast specifických nastavení vybrané hry. Implementace tohoto panelu je pro větší variabilitu ovládacích prvků a tím pádem intuitivní a komfortní zadávání potřebných hodnot přenechána na jednotlivých hrách. Jedna z metod rozhraní hry vrací instanci abstraktní třídy GameSpecificSettingsPanel, která je potomkem JPanelu. Tvůrce hry tak musí vytvořit svou třídu, kterou zdědí z GameSpecificSettingsPanelu, umístí na ní veškeré potřebné komponenty na zadávání hodnot a zajistí implementaci metody getGameSpecificSettingsMap(). Ta zajišťuje, že GameControl modul z panelu specifických nastavení bude schopen získat hodnoty nastavených parametrů a předat je hře při jejím spuštění.

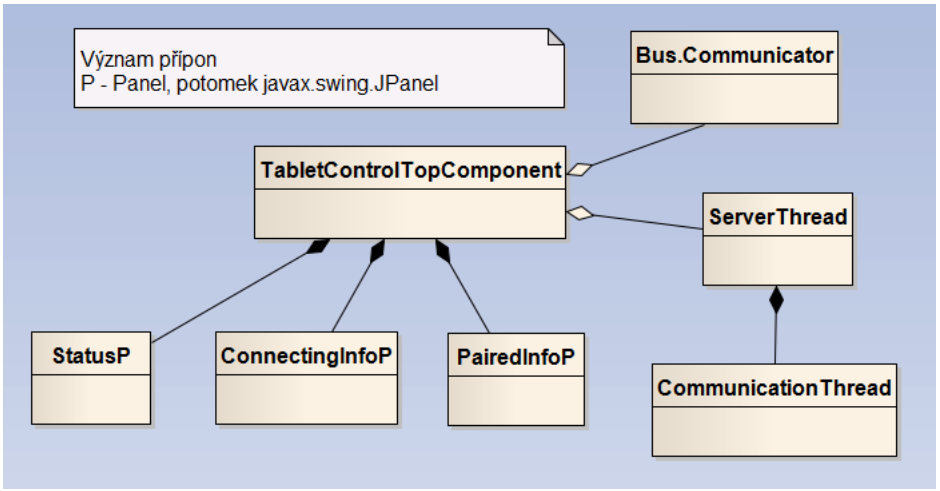
Třetí částí modulu je panel pro úpravu nastavení parametrů kanálů. Pokud bylo v modulu ClientControl vybráno kolo, kterému ještě nebyl nahrán průběh, nastavení kanálů bude přednastaveno na hodnoty odpovídající nastavení kanálů z varianty nastavení parametrů sezení vybraného kola. Tyto hodnoty se dají měnit uživatelem a jsou pak předány hře při jejím spuštění. Není-li kolo vybráno, jsou zobrazeny výchozí hodnoty a uživatel musí nastavení provést sám. K dispozici jsou nastavení pro čtyři kanály, každá hra si však sama určí, které používá jako své vstupy a na základě toho pak mohou být některé kanály pro některé hry zakázané.

Polední částí modulu je oblast informací z ostatních panelů. Jsou zde dva informační panely, jeden obsahuje data o právě vybraném kole v modulu ClientControl a druhý stučné informace o připojeném zařízení v modulu DeviceSettings.

6.2.9 TabletControl

Grafické rozhraní je jako u ostatních modulů zajištěno pomocí potomka TopComponent, do kterého jsou dynamicky přidávány další panely s potřebnými grafickými prvky. Modul má tři možné stavy, pro každý stav je je předdefinováno jiné rozložení panelů.

Obrázek 17 Class diagram pro modul TabletControl



Ve stavu „SOCKET_CLOSED“ je v modulu pouze pole informující o stavu modulu a tlačítko na spuštění připojovací fáze. Stisknutím tlačítka se modul převede do stavu „OPENED_AND_WAITING“. Namísto původního tlačítka na otevření socketu se zobrazí panel s tlačítkem pro zavření socketu, informacemi o síťové adrese a portu,

kde aplikace má otevřený socket, a na pozadí se spustí serverová část komunikace s mobilním zařízením. Ta je implementována klasickou metodou Java socketů, kde serverová aplikace na zadaném portu naslouchá a čeká na připojení klienta. Po připojení klienta se provede handshake a komunikace pak probíhá v separovaném vlákně.

Připojením klienta se modul přepne do stavu „PAIRED“, panel s informacemi o socketu je nahrazen panelem informujícím o stavu připojení s mobilním zařízením. Když tablet informuje server, že má navázáno spojení s inteligentním robotem, je finálně modul připraven pro použití (volání metody isTabletReady() bude vracet True a při volání metody rozhraní ISphero vrátí stav OutputDeviceState.READY).

Po spuštění hry, která má jako výstupní zařízení Sphero, pak data vysílaná Hubem jako výsledky výpočtů nad přijatými daty ze vstupního zařízení jsou pomocí modulu TabletControl přesměrovány na mobilní zařízení, které je dále přepošle robotovi. Ten se na základě příkazů může pohybovat, točit a měnit barvy.

V případě odpojení tabletu nebo výpadku spojení se modul přepne zpět do stavu „OPENED_AND_WAITING“, místo panelu PairedInfoP se vloží ConnectingInfoP a server znovu bude čekat na připojení mobilního zařízení.

6.2.10 Moduly her

Jak bylo zmíněno v analytické části, nejedná se o moduly ve smyslu modulů platformy NetBeans, ale pouze o samostatné jednotky jednotlivých her. Hry jsou naprosto nezávislé na všech ostatních částí aplikace a komunikují pouze přes své rozhraní. Moduly her v aktuálním architektonickém návrhu nevyužívají komunikátor a tím pádem ani intermodulární komunikační protokol. Potřebná data jako signály ze vstupního zařízení, které zpracovává a rozesílá modul Hub ze sběrnice čte modul GameControl a přeposílá je běžící hře přes její interface.

Každá hra musí obsahovat třídu Core, která implementuje interface IGame. Při vybrání hry v modulu GameControl se na základě informací o hře v databázi provede vytvoření instance třídy Core z balíčku hry, tato instance je uložena, a je pak využívána pro veškerou komunikaci s hrou.

Na základě rozhraní každá hra umí přijímat mapu specifických nastavení, nastavení kanálů, nastavení pásem. Všechna nastavení jsou jí modulem GameControl poslána před jejím spuštěním. Další metody jsou pro spuštění a zastavení hry.

Hry jsou zodpovědné za své grafické rozhraní samy. Pokud hra má jako výstupní zařízení počítačovou obrazovku, sama si vytvoří okno, ve kterém poběží. Zprávy příkazů pro jiná výstupní zařízení pak předává modulu GameControl, který je přes intermodulární protokol odesílá na rozhraní výstupního zařízení.

7 Použití aplikace a ukázky GUI

Celkem je případů užití systému opravdu obrovské množství, rozhodl jsem se proto popsat jen několik z těch nejčastěji používaných a doplnit je screenshoty aplikace. Protože se jedná o případy, pro které byl celý systém navržen, patří zároveň mezi ty nejsložitější.

Aplikace je v aktuální verzi optimalizována pro FullHD (1920x1080) rozlišení a pro běh v maximalizovaném okně.

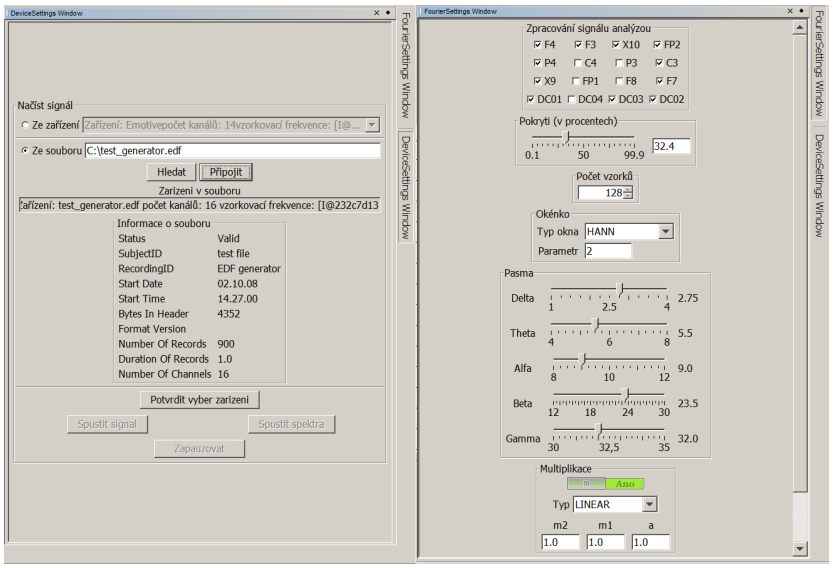
7.0.11 Zobrazení dat ze vstupního zařízení a výsledků analýzy

Je potřeba začít připojením vstupního zařízení, což se dělá přes modul DeviceSettings. Jeho rozhraní je vidět na obrázku 18 v levé části. Vybere se buď zařízení přímo připojené k počítači nebo z již nahraného souboru, pokud je zvolena cesta souboru, je potřeba vyplnit cestu a soubor připojit. Pak je potřeba potvrdit výběr zařízení.

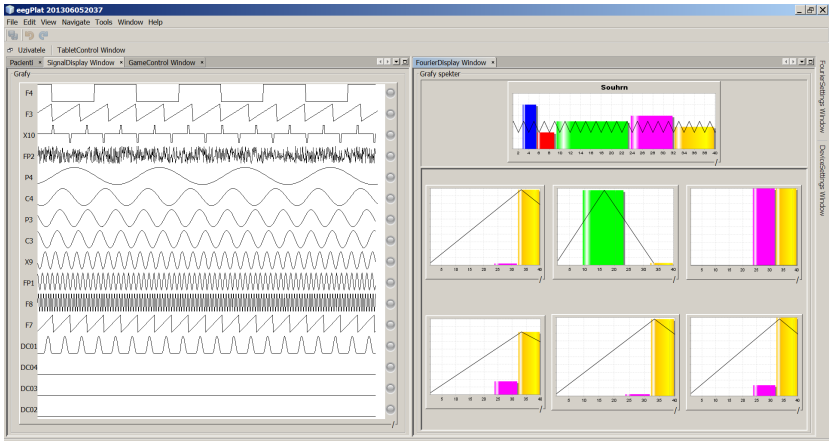
Když máme připojené zařízení, je potřeba v modulu FourierSettings nastavit parametry analýzy. Rozhraní modulu Fourier settings je vidět v pravé části obrázku 18. Po nastavení požadovaných hodnot je třeba nastavení potvrdit. Dělá se to tlačítkem umístěným ve spodní části modulu (na obrázku tlačítko není vidět, bylo by potřeba odscrollovat níže).

V modulu DeviceSettings je pak možné spustit přijímání signálů ze zařízení a jejich zpracování analýzou pomocí příslušných tlačítek. V modulech SignalDisplay a FourierDisplay se pak data kreslí. Kompaktní režim zobrazení dat, který je vytvořen umístěním obou modulů vedle sebe na obrazovku, je vidět na obrázku 19.

Obrázek 18 Rozhraní modulů DeviceSettings a FourierSettings



Obrázek 19 Rozhraní modulů SignalDisplay a FourierDisplay



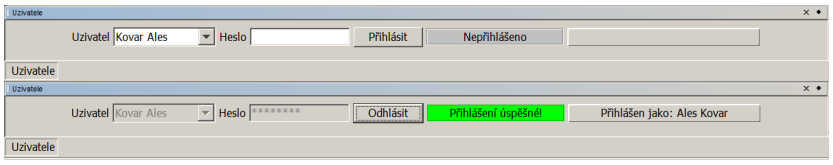
7.0.12 Založení nového klienta a správa jeho diagnóz, terapií a sezení

Pro fungování administrativního modulu je potřeba se přihlásit do aplikace pomocí modulu UserControl. Rozhraní modulu UserControl je na obrázku 20, zachyceny jsou oba stavy – pokud není uživatel přihlášen (nahoře) a po příhlášení (dole).

Přihlášením se odemkne modul ClientControl. Na obrázku 21 je vidět jeho rozhraní po vybrání možnosti vytvoření nového klienta. Po vyplnění formuláře a jeho odeslání tlačítkem Vytvořit se modul automaticky přepne do zobrazení detailů klienta. Jak takové zobrazení běžně vypadá je vidět na obrázku 22. Zde lze v příslušných blocích přidávat nebo upravovat diagnózy a terapie klienta.

Pro správu sezení se stačí přepnout do sekce správy sezení. Lze to udělat pomocí Časové osy v horní části modulu, vybráním nějakého sezení ze seznamu sezení klienta nebo pomocí tlačítka na vytvoření nového sezení. Náhled sekce správy sezení s vybraným sezením (se zobrazením jeho detailů) je na obrázku 23.

Obrázek 20 Rozhraní modulu UserControl v nepřihlášeném i přihlášeném stavu



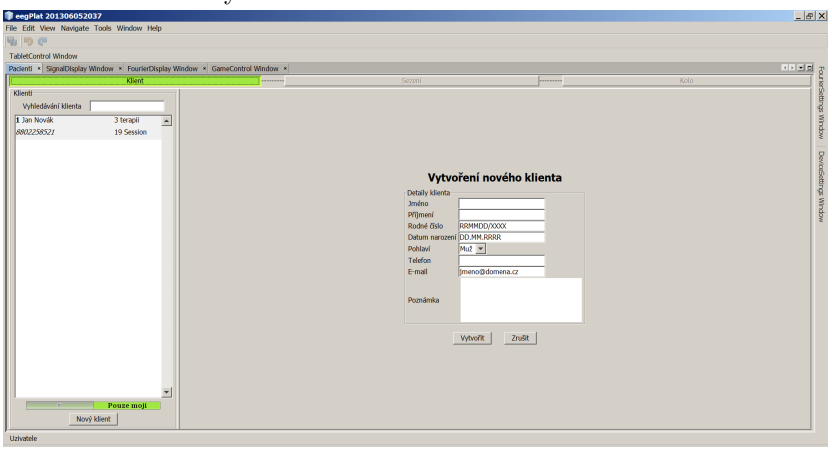
7.0.13 Spuštění hry a nahrání průběhu jednoho kola

Pro nahrání a uložení průběhu jednoho kola je potřeba vybrat nenahrané kolo, připojit vstupní zařízení, vybrat a nakonfigurovat jednu z her a spustit jí.

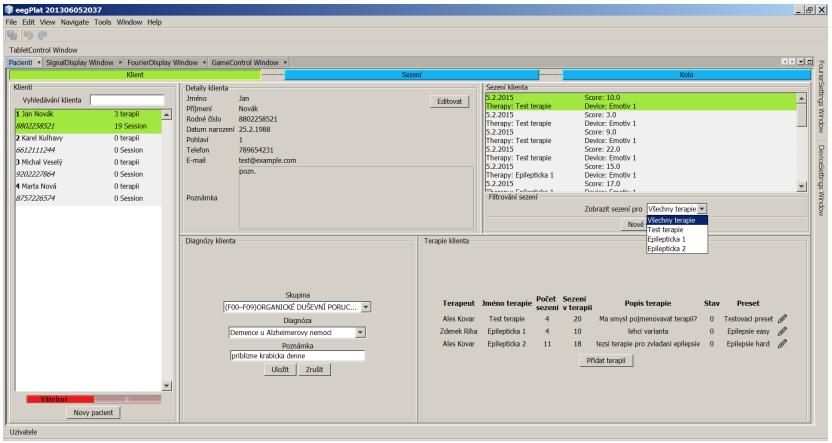
Které kolo se bude nahrávat se vybírá přes modul ClientControl. Stejně jako při scénáři vytváření a správy klientů je potřeba se přihlásit přes modul UserControl. Pak lze vybrat klienta, některé z jeho sezení a v rámci sezení pak kolo. Pokud toto kolo již má nahraná data, modul správy her uživatele na tuto skutečnost upozorní červeným podbarvením panelu s informacemi o vybraném kole tak, jak je vidět na obrázku 25. Pokud kolo není nahráno, modul ClientControl vyzve uživatele k nahrání kola, pokud nahráno je, zobrazí výsledky klienta v průběhu kola jako graf (obrázek 24).

Jak se připojuje vstupní zařízení je popsáno ve scénáři zobrazení zpracovávaných dat. Modul správy her o připojeném zařízení uživatele informuje podobným způsobem jako

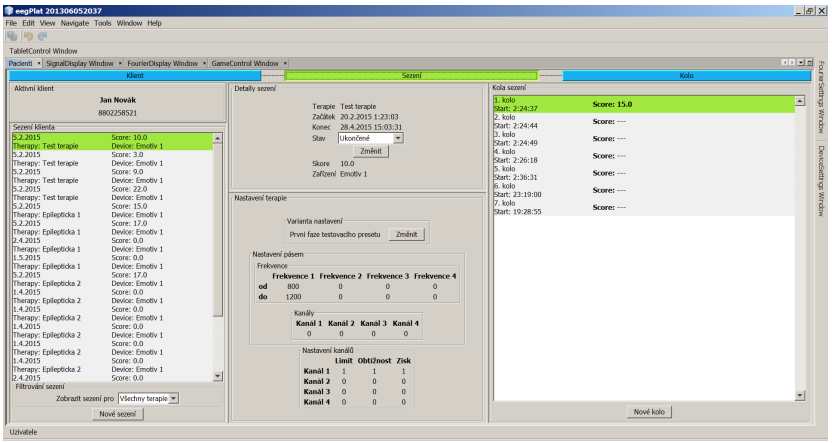
Obrázek 21 Vytvoření nového klienta v modulu ClientControl



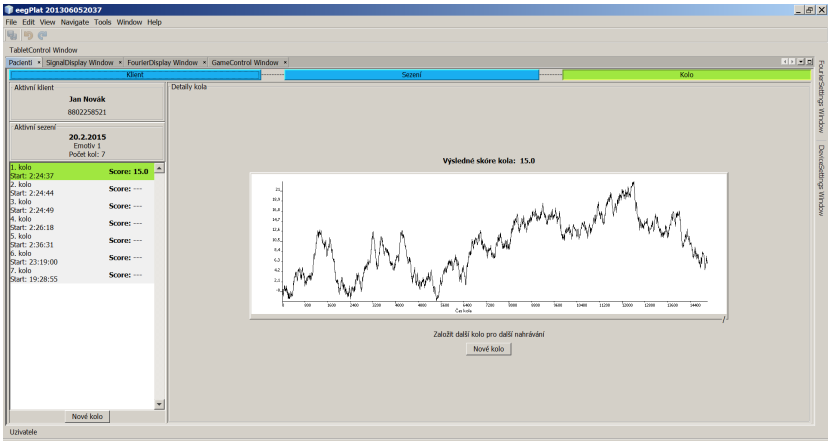
Obrázek 22 Rozhraní modulu ClientControl v sekci správy klientů



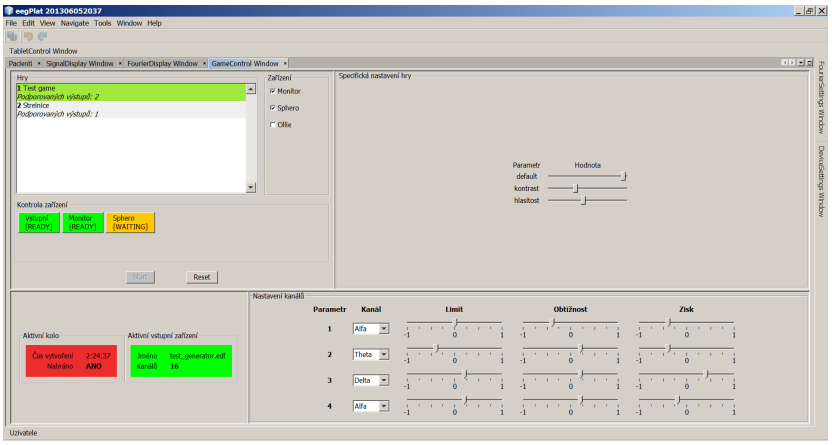
Obrázek 23 Rozhraní modulu ClientControl v sekci správy sezení



Obrázek 24 Rozhraní modulu ClientControl v sekci správy kol



Obrázek 25 Rozhraní modulu GameControl



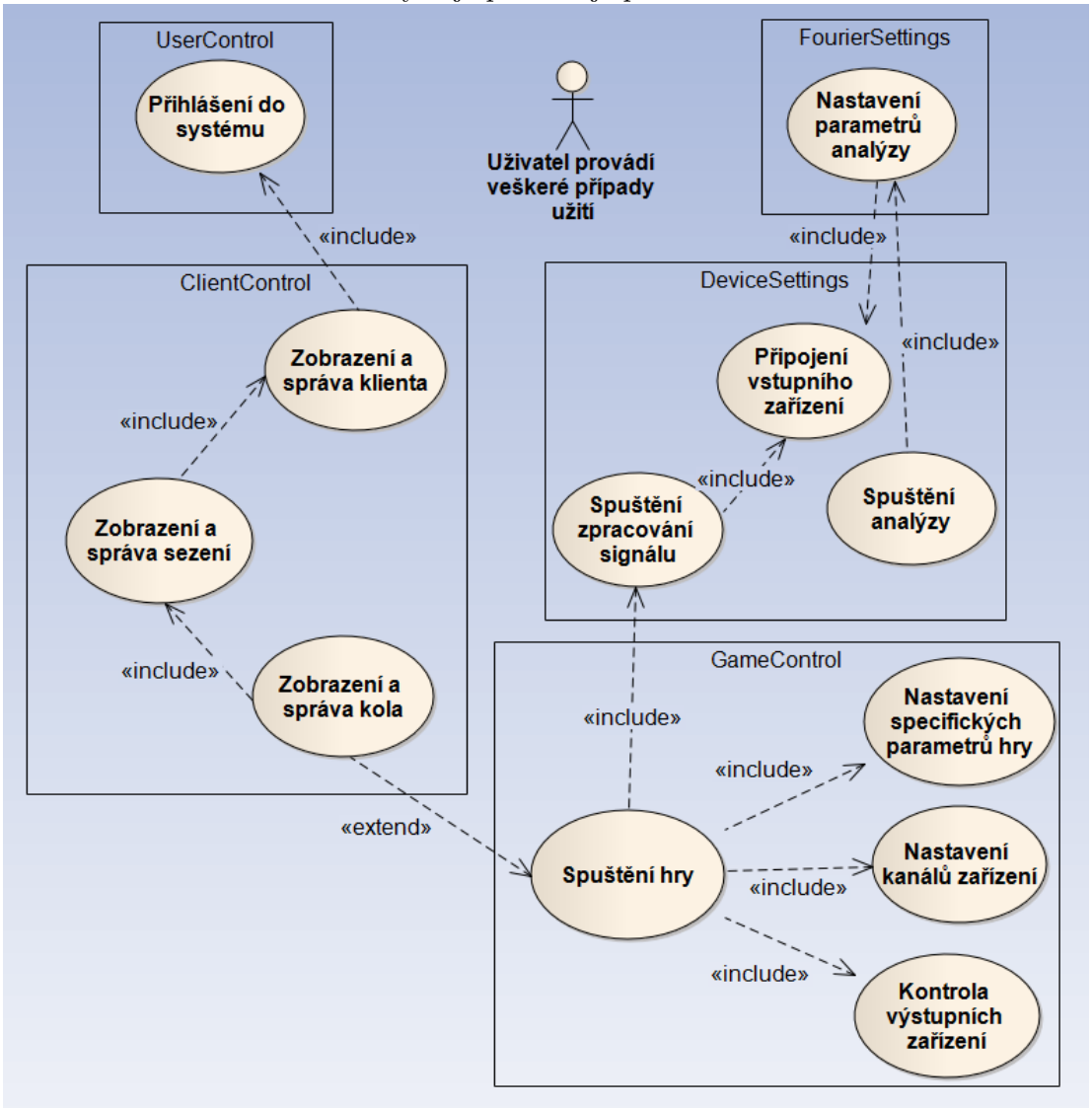
o vybraném kole – v levé spodní části modulu je informační panel, který aktualizuje své zobrazení podle stavu vstupního zařízení. Aby bylo vstupní zařízení připravené na spuštění hry, musí být spuštěný příjem signálů.

Konfigurace hry se provádí pouze v rámci modulu GameControl (obrázek 25). Je potřeba vybrat hru ze seznamu, zvolit požadovaná výstupní zařízení ze seznamu a nastavit její specifické parametry. Oblast kontroly zařízení zobrazuje kontrolky podle stavu zařízení, ke kterému patří. Když jsou všechny zelené (ve stavu READY), je možné spustit hru. Hra běží dvě minuty od spuštění (délka jednoho kola) a zaznamenávají se údaje o úspěšnosti klienta. Po skončení kola je v modulu ClientControl možné zobrazit úspěšnost klienta v jeho průběhu (obrázek 24).

Protože se jedná o scénář spojující funkčnost téměř všech modulů aplikace, pro přehlednost je na obrázku 26 diagram případů užití potřebných pro tento scénář. Dílčí případy užití se dají pak rozložit jak je rozebráno v kapitole 5.

Obrázek 26 Stručný přehled případů užití celé aplikace

Vypsána je pouze kostra případů užití pro typické použití aplikace během každodenní práce terapeuta s klienty. Jednotlivé případy užití jsou rozděleny do modulů, ve kterých je potřeba je provádět.



8 Testování a budoucí vývoj

8.1 Testování

Protože se jedná o uživatelské rozhraní, základní metoda testování zdrojového kódu – tím mám na mysli unit testy – nemá příliš smysl. Proto unit testy nejsou v projektu implementovány.

Stejně tak integrační testování a testování rozhraní komponent nepřináší potřebné výsledky. Integrační testování se vzhledem k modularitě systému nabízí, ale je potřeba si uvědomit, že jednotlivé moduly spolu vlastně napřímo nekomunikují. Komunikace mezi nimi probíhá výhradně po intermodulárním komunikačním protolu.

Funkcionalita intermodulárního komunikačního protokolu byla během jeho stavění opakovaně testována pečlivým krokováním algoritmu a byla tak odladěna drtivá většina chyb. Od dokončení a odtestování protokolu (při jeho správném použití) za celou dobu vývoje ostatních modulů nevznikla jeho vinou jediná chyba.

Testování samotného uživatelského rozhraní bylo prováděno manuální cestou, vytvoření automatizovaných testů by zabralo obrovské množství času. Navíc díky modifikovatelnosti celého uživatelského rozhraní a možnosti přeskupovat moduly v aplikaci je použití nahrávaných maker pro testování rozhraní zásadně složitější a muselo by být nahráváno vícero testovacích cyklů pro každý testovací scénář.

Otestovány byly veškeré případy použití popsané v sekci analytické dokumentace a byla ověřena jejich funkčnost.

8.2 Budoucí vývoj

Aktuální verze aplikace poskytuje základní funkcionalitu a odpovídá stanoveným požadavkům, nejedná se však o finální produkt a její vývoj bude nadále pokračovat. Do budoucna je už naplánovaná celá řada dalších rozšíření, která budou probíhat v rámci grantu Medialab [20]. Rád bych tu zmínil alespoň některé z nich.

8.2.1 Administrativní modul

Pro správu obsahu té části databáze, do které nemají přístup běžní uživatelé a terapeuti je plánován Admin modul. Umožní rychlé a jednoduché změny hodnot v databázových tabulkách nastavení pásem a kanálu, změny v presetech terapii, přídávání a měnění jejich variant a přiřazování jednotlivých presetů k diagnózám.

8.2.2 Automatické instalování nových aktualizací

Využitím update centra platformy NetBeans lze zajistit automatickou distribuci a instalaci drobných aktualizací jednotlivých modulů aplikace. Při spuštění se systém sám připojí na centrální server, zkontroluje dostupnost aktualizací a pokud jsou nějaké k dispozici, stáhne je do počítače a provede jejich instalaci.

8.2.3 Automatické hledání kompatibilních aplikací v síti

Pro moduly komunikující po síti je před jejich používáním potřeba provést spárování se vzdálenou stranou. V současné verzi se toto párování provádí manuálně – opsáním poskytnutých síťových adres z jednoho zařízení do druhého. V budoucnu by měl systém podporovat automatické vyhledávání zařízení s běžící aplikací na lokální síti. To by vedlo k tomu, že by se zařízení párovala pouze vybráním požadovaného protějšku z poskytnutého seznamu. To by vedlo k zásadnímu urychlení párování a pro uživatele by to bylo mnohem komfortnější.

8.2.4 GUI pro mobilní zařízení

Díky architektonickému návrhu intermodulárního komunikačního protokolu je v budoucnu možné vytvořit lightweight verzi stávajícího grafického rozhraní pro mobilní zařízení (tablety nebo dokonce telefony). Mobilní GUI by obsahovalo pouze omezenou funkčnost, ale výhledově by mělo využití třeba jako možnost dálkového spouštění her v případě, kdy by výstupním zařízením bylo něco, co není možné provozovat v blízkosti stolního počítače, na kterém by běžela hlavní aplikace.

8.2.5 Centrální server, databáze a zálohování dat

Aplikace v každém terapeutickém centru má v současnosti svou vlastní databázi. V plánu je všechny tyto databáze připojit k centrální databázi, do které by se nahraná data z center v pravidelných intervalech zálohovala.

Přihlašování aplikací k centrálnímu serveru by navíc mohlo poskytnout další řadu možností, převážně statistických. Šlo by snadno zjistit kde aplikace běží, v jakém počtu, kolika terapeuty jsou využívány, kolik klientů se pravidelně zúčastňuje terapií a zdali prokazují nějaká zlepšení. Tyto statistiky, samozřejmě v naprosto zanonymizované formě, pak mohou přinášet zajímavá data potřebná pro další rozvoj a vylepšování celého systému.

8.2.6 Vzdálené GUI

Spolu s existencí centrálního serveru se pak nabízí možnost vzdáleného uživatelského rozhraní – pravděpodobně v podobě webového rozhraní.

Aplikace by mohla zpřístupnit část své funkcionality pomocí webových stránek. Ty by nemusely být veřejně dostupné ale šlo by s nimi komunikovat pouze prostředníctvím centrální serveru. Byla by tak zajištěna dostatečná bezpečnost komunikace a umožňovalo by to vzdálené řešení běžných administrativních problémů, které časem při běhu aplikace budou bez pochyb vznikat.

8.2.7 Pokročilé zobrazování signálů a výsledků analýz

Ostatní biofeedbackové a neurofeedbackové softwary obsahují různé zajímavé způsoby zobrazování snímaných dat. Do aplikace je v plánu přidat další moduly, které budou stejně jako moduly SignalDisplay nebo FourierDisplay odebírat příchozí zprávy s daty zpracovaných i nezpracovaných signálů a provádět jejich pokročilou vizualizaci.

8.2.8 Automatizované testy GUI a testy jádra

Jak neustále roste složitost celé aplikace a objem zdrojových kódů, je potřeba vytvoření robustního systému automatizovaných testů čím dál naléhavejší. S přidáváním nové

funkcionality je velmi pravděpodobné, že se do stávajícího kódu zanesou nějaké nové chyby a jediný způsob jak tyto chyby efektivně odhalovat je extenzivní automatické testování.

8.3 Známé problémy

Testováním došlo také k objevení několika chyb v aplikaci, které jsou z různých důvodů zatím neopravené.

Prvním problémem je nekonzistence zobrazení zaškrtávacích políček na panelu výběru výstupních zařízení v modulu GameControl s jejich datovým modelem. Při některých bězích aplikace se stane, že po výběru hry ze seznamu nedojde k zaškrtnutí příslušných checkboxů na panelu výběru výstupních zařízení. Všechny potřebné metody se zavolají, proběhnou bez chyb (zjištěno krokováním aplikace) a dokonce při výpisu hodnot jednotlivých komponent se vše tváří v pořádku. Jen se zaškrtávátka špatně zobrazují a špatně pak reagují na uživatelský vstup.

Dalším problémem je nastavování stavu modulu UserControl. V některých případech se při přihlášení uživatele do aplikace stane, že modul zůstane ve stavu nepřihlášeného uživatele. Přihlášení ale proběhne v pořádku, ostatní moduly fungují bez problémů, jen samotný UserControl na přihlášení nereaguje. Při krokování aplikace tento problém nenastává a modul funguje v pořádku, v běžném běhu to zhruba v polovině případů nefunguje.

Oba tyto problémy jsou zřejmě způsobeny nějakým vnitřním problémem v platformě NetBeans a tak nezbývá než vyřešit problémovou funkcionalitu jiným způsobem nebo migrovat na novější verzi NetBeans platformy.

9 Závěr

Cílem mé práce bylo vytvořit komplexní grafické uživatelské rozhraní nad existujícím jádrem aplikace.

Teoretická část práce mi poskytla východisko pro zpracování praktické části. Popsal jsem východiska neurofeedbacku – mozek, jeho projevy a jejich měření a pak už jsem konkrétně pracoval s neurofeedbackem jako takovým – definicí, využitím a přínosem. Dále jsem podrobně popsal modulární architekturu aplikace s důrazem na intermodulární komunikační protokol, který moduly propojuje. Byly popsány technologie použité k implementaci a byly dopodrobna rozebrány případy užití jednotlivých modulů.

V praktické části už jsem se zabýval konkrétně implemetací modulů a ukázkami uživatelského rozhraní. Popsal jsem i testování rozhraní a uvedl další možnosti rozvoje.

Přínosem mé práce je nejen vytvoření uživatelského rozhraní, ale také podíl na projektu, který má obrovský potenciál. Přináší základní spojení administrativního softwaru pro správu klientů se softwarem pro řízení neurofeedbackových terapií. Projekt v této fázi představuje mezikrok, který však přináší možnosti do budoucna. Modulární architektura spolu s robustním návrhem intermodulárního komunikačního protokolu umožňuje snadné rozšiřování aplikace o nové moduly přínášející novou funkcionalitu, možnost vytvářet řadu dalších her a přidávat další propojení s všemožnými věcmi – ať se jedná o inteligentní roboty, nová zařízení na snímání dat čí další části uživatelského rozhraní v podobě modulů s pokročilou vizualizací přijímaných dat nebo lightweight aplikací pro mobilní zařízení, architektura je navržená tak, že to zvládne. Mnou vytvořené uživatelské rozhraní je přehledné, intuitivní a snaží se uživateli poskytnout při ovládání aplikace co nejvyšší komfort možností si pracovní prostředí co nejvíce přizpůsobit svým potřebám. Cíl mé práce byl naplněn.

Příloha A

Obsah přiloženého CD

Cesta	**Popis**
install	Adresář instalátorů
install/eegplat-linux.sh	instalátor pro Linux
install/eegplat-macosx.tgz	instalátor pro Mac
install/eegplat-solaris.sh	instalátor pro Solaris
install/eegplat-windows.exe	instalátor pro Windows
zeman-dipl.pdf	PDF soubor s textem této práce

Literatura

[1] CHERRY Kendra. *The Anatomy of the Brain. About Education.* URL: http://psychology.about.com/od/biopsychology/ss/brainstructure.htm#step-heading (cit. 06. 05. 2015).

[2] LUNGOVÁ Vlasta. *Stavba a funkce lidského mozku. E-learningová podpora mezioborové integrace výuky tématu vědomí na UP Olomouc.* URL: http://pfyziollfup.upol.cz/castwiki/?p=3265 (cit. 06. 05. 2015).

[3] *Brain Lobes.* URL: http://www.mayoclinic.org/brain-lobes/img-20008887 (cit. 06. 05. 2015).

[4] *Neurony a gliové buňky.* URL: http://www.poranenimozku.cz/fakta-o-mozku/mozek-na-bunecne-urovni/neurony-a-gliove-bunky.html (cit. 06. 05. 2015).

[5] *What are Brainwaves?* URL: http://www.brainworksneurotherapy.com/what-are-brainwaves (cit. 06. 05. 2015).

[6] *Electroencephalography.* URL: http://en.wikipedia.org/wiki/Electroencephalography (cit. 06. 05. 2015).

[7] MALMIVUO Jaakko a Robert PLONSEY. *Bioelectromagnetism: principles and applications of bioelectric and biomagnetic fields.* 1. vyd. New York: Oxford University Press, 1995. 482 s. ISBN: 01-950-5823-2.

[8] COBEN Robert a EVANS James R. *Neurofeedback and neuromodulation techniques and applications.* 1. vyd. 2011. 431 s. ISBN: 0123822351.

[9] *Intro to Neurofeedback. EEG Education and Research Inc.* URL: http://www.eegspectrum.com/intro-to-neurofeedback (cit. 06. 05. 2015).

[10] *Neurofeedback for ADHD: Significant, Lasting Improvement.* URL: http://www.medscape.com/viewarticle/821113 (cit. 10. 05. 2015).

[11] *Pro-spiro, prodej a servis zdravotní techniky.* URL: http://www.pro-spiro.cz/data/biosoft.php (cit. 06. 05. 2015).

[12] *Medical devices for neurology, cardiology and rehabilitation.* URL: http://www.neurosoft.com/en/ (cit. 06. 05. 2015).

[13] *EEGStudio Software: EEG and ERP acquisition software.* URL: http://www.mitsar-medical.com/eeg-software/erp-software/ (cit. 06. 05. 2015).

[14] *About the Java Technology.* URL: https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html (cit. 06. 05. 2015).

[15] *What is Java and why do I need it?* URL: https://www.java.com/en/download/faq/whatis%7B%5C_%7Djava.xml (cit. 06. 05. 2015).

[16] *NetBeans IDE – NetBeans Rich-Client Platform Development (RCP).* URL: https://netbeans.org/features/platform/ (cit. 06. 05. 2015).

[17] *MySQL :: The world's most popular open source database.* URL: http://www.mysql.com/ (cit. 06. 05. 2015).

[18] Oracle and/or its affiliates. “Top 10 Reasons to use MySQL as an Embedded Database”. In: *A MySQL Technical White Paper* (2014).

[19] *State Design Pattern*. URL: `https://sourcemaking.com/design_patterns/state` (cit. 06. 05. 2015).

[20] *ČVUT Media Lab: ČVUT Media Lab*. URL: `http://www.cvutmedialab.cz/` (cit. 06. 05. 2015).