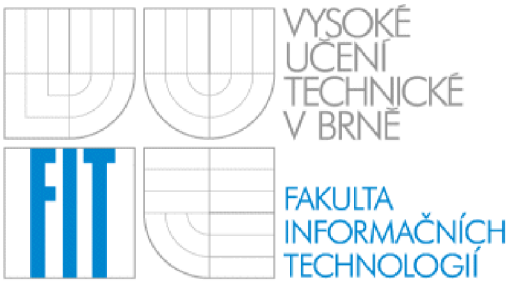
Vysoké učení technické v Brně
Fakulta informačních technologií



Diplomová práce

Brno 2007 **Jan Kněžík**

Vysoké učení technické v Brně – Fakulta informačních technologií

Akademický rok 2006/2007

Zadání diplomové práce

Řešitel: **Kněžík Jan**
Obor: Výpočetní technika a informatika
Téma: **EEG biofeedback rozhraní lidského mozku a počítače**

Pokyny:

1. Na základě již řešeného stejnojmenného semestrálního projektu vypracujte specifikaci konkrétního systému pro EEG biofeedback.
2. Navrhněte programové vybavení pro tento systém s důrazem na objektově orientované řešení pomocí návrhových vzorů (design patterns).
3. Proveďte implementaci daného systému pomocí jazyka Java.
4. Navrhněte možnosti ovládání směru pomocí EEG zařízení.
5. Navrhněte možná využití výsledků Vaší práce a možnosti dalšího rozvoje.

Literatura:

- http://www.eegspectrum.com/
- www.biof.com

Vedoucí: **Marušinec Jaromír, Ing., Ph.D.**, CVIS VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 24. ledna 2007

Prohlašuji, že jsem tuto diplomovou práci vypracoval sám pod vedením Ing. Jaromíra Marušince, Ph.D.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Brně 24. 1. 2007 .....

Abstrakt

Tato práce se zabývá rozhraním EEG biofeedbacku mezi lidským mozkem a počítačem a jeho konkrétní implementací v jazyce Java. Tento systém je postaven na základě počítače uskutečňujícího biologickou zpětnou vazbu (biofeedback) a elektroencefalografie (EEG), pomocí níž je prováděno snímání stavů mozku uživatele. Tímto způsobem je možné mozek efektivně trénovat a dosáhnout lepšího soustředění, odstranění poruch spánku a učení. Dále je zde obsažen návrh ovládání směru počítačové myši pomocí EEG zařízení, u kterého je člověk schopen pomocí kmitočtu mozkových vln řídit směr pohybu kurzoru po obrazovce. Motivací k řešení tohoto problému je snaha pomoci postiženým lidem komunikovat s okolním světem. Úvod práce obsahuje základní fakta o lidském mozku, elektroencefalografii a EEG biofeedbacku. Následují kapitoly se zabývají specifikací požadavků na vyvíjenou aplikaci, jejím návrhem a popisem vlastní implementace. Závěrečná část je věnována systémům BCI (Brain-Computer Interface) a návrhu ovládání počítače pomocí EEG zařízení se zhodnocením dosažených výsledků.

Klíčová slova:

EEG, BCI, biofeedback, elektroencefalografie, mozek, rozhraní, počítač, zpětná vazba, biopotenciály, ovládání směru, myš, ModularEEG

Abstract

This master thesis dwells on EEG biofeedback (also called Neurofeedback) interface of human brain and the computer and its concrete realization in Java programming language. This system is founded on the basis of the computer, which is accomplishing biological feedback (biofeedback) and the electroencephalography (EEG) by helping that state's scanning of user's brain is realized. By this way is possible to practise the human brain effectively to achieve better concentration, the elimination of sleeping and learning deficiency. Hereafter is the suggestion of direction control of computer mouse by EEG device incorporated, which makes it possible for the man to regulate the direction of the cursor's movement on the screen by the frequency of brain's oscillation. The motivation for solution of this problem is the effort to help to handicapped people to communicate with surrounding world. The introduction of this paper contains the basic facts about human brain, electroencephalography and EEG biofeedback. The following chapters dwell on the specification of claims to developed application, its suggestion and description of actual realization. The final part relates to the BCI (Brain-Computer Interface) systems and suggestion of computer's control by EEG appliance with evaluation of attained results.

Keywords:

EEG, BCI, biofeedback, electroencephalography, brain, interface, computer, feedback, biopotentials, direction control, mouse, ModularEEG

Obsah

1. Úvod

Tato práce patří do širšího oboru informačních technologií nazvaného signály a systémy. Zabývá se jednak softwarovými prostředky pro podporu práce s biologickou zpětnou vazbou realizovanou pomocí EEG zřízení a osobního počítače (EEG biofeedback) a jejich implementací, dále pak pokračuje tématem rozhraní lidského mozku a počítače, které víceméně souvisí s tématem předchozím a představuje určité vývojové pokračování a rozšíření dané problematiky.

Od 60. let probíhá v oblasti neurověd (Neurosience) intenzivní výzkum, který nám přinesl řadu nových poznatků o fungování lidského mozku. Jedním z výsledků experimentů je vynález biologické zpětné vazby na bázi elektroencefalografie (EEG biofeedback), která pracuje s elektrickými impulsy vysílanými mozkem a získává z nich informaci o jeho stavu. Cílem je dosáhnou co nejlepšího „vyladění“ mozkových vln, které zajistí podle nastavení systému uživateli schopnost zvýšeného soustředění, relaxace nebo navození spánku. Používaní systémů vytvořených na základě biologické zpětné vazby je tak velice efektivním prostředkem k ovládnutí lidského mozku a tím přeneseně i vlastní mysli.

Donedávna byl vývoj a používání těchto systémů z důvodu poměrně vysoké ceny EEG zařízení (více jak 1000 $) omezen na okruh lidí z vědeckého a lékařského prostředí. Nyní je na internetu možné najít několik skupin, vyvíjejících hardware pro EEG nebo přímo pro EEG biofeedback pod licencí GPL nebo jako public domain. Hlavním důvodem k řešení tohoto projektu bylo to, že v současné době chybí open–source software pro EEG biofeedback, který by se mohl měřit s profesionálními systémy pro klinickou praxi. K výše zmiňovaným zařízením existují především aplikace vizualizující měřené signály, ale skoro žádná pro praktický tréning mozkových vln prostřednictvím hry. Tato práce si klade za cíl tuto mezeru zaplnit a poskytnout široké skupině odborníků a nadšenců použitelný nástroj za přijatelnou cenu.

Mezi klasická vstupní periferní zařízení, sloužící k ovládání počítače, vždy patřila sada tlačítek, ze které se později postupně vyvinula klávesnice. S příchodem grafických uživatelských rozhraní se tato množina rozšířila také o způsob ovládání pomocí počítačové myši nebo světelného pera. Pro uživatele počítače je tedy velmi důležitá dovednost v používání výše uvedených periferní zařízení, jinak je možnost používaní počítačů prakticky vyloučena. Mnoho postižených lidí to ale nedokáže a je ochuzeno o možnost aktivně pracovat na počítači a jeho prostřednictvím komunikovat s okolním světem. V minulosti již byla vyvinuta celá řada systémů pro pomoc postiženým lidem, například ovládání počítače pomocí pohybů očí. Nemohu zde také nezmínit příklad britského astrofyzika Stephena Hawkinga, trpícího amyotrofickou laterální sklerózou, který používá hlasový syntetizátor ovládaný pomocí pohybů lícního svalu. V současné době se ovšem do popředí zájmu dostávají rozhraní typu BCI (Brain–Computer Interface, rozhraní lidského mozku a počítače), která se snaží o ovládání počítače pomocí myšlení uživatele. K tomu podobně jako EEG biofeedback používají elektroencefalografii a ze získaných signálů se snaží vydedukovat požadovaný povel uživatele. V této práci se zabývám návrhem jednoduchého rozhraní, které by umělo nahradit počítačovou myš při ovládání pohybu kurzoru po obrazovce.

Stručný přehled obsahu jednotlivých kapitol obsažených v této práci:

Druhá kapitola má název Anatomie a funkce lidského mozku. Obsahuje seznámení se stavbou nervové soustavy člověka a lidského mozku.

Třetí kapitola je o elektroencefalografii. Zabývá se shrnutím základních principů této metody a popisem vlastností používaných zařízení.

Ve čtvrté kapitole je pojednáno o systémech s biologickou zpětnou vazbou a jejich kombinací s EEG (EEG biofeedback).

Pátá kapitola obsahuje specifikaci navrhovaného software pro EEG biofeedback a vytýčené cíle projektu, kterých by se mělo v průběhu času dosáhnou.

V šesté kapitole čtenář nalezne zprávu o návrhu příslušného programového vybavení v jazyce Java.

Sedmá kapitola se zabývá implementací navrženého software s důrazem na popis problémů vzniklých při řešení

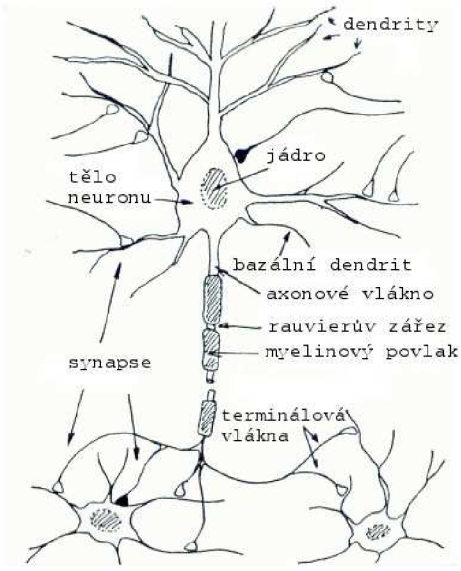
Obsahem osmé kapitoly je teorie rozhraní lidského mozku a počítače, návrh ovládání směru kurzoru pomocí EEG zařízení a prezentace výsledků při experimentech s tímto rozhraním.

Devátá kapitola shrnuje vytyčené cíle a dosažené výsledky při řešení této práce a nastiňuje možnosti dalšího rozvoje.

Druhá, třetí a čtvrtá kapitola je v pozměněné podobě převzata ze stejnojmenného semestrálního projektu řešeného autorem [1].

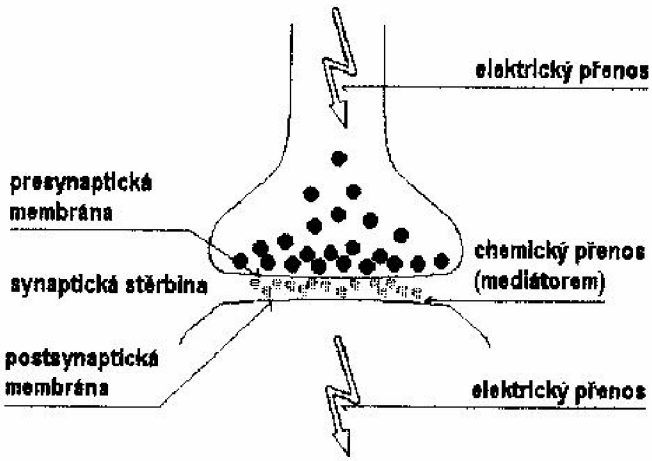
2. Anatomie a funkce lidského mozku

Lidský mozek [2][3] je nesmírně složitá struktura. Je to v podstatě nejsložitější známá struktura vůbec. Obsahuje nervové buňky - neurony a jejich vlákna (axony), jimiž jsou mezi sebou vzájemně propojeny. Počet nervových buněk je obrovský (odhadem 10^{11} neuronů), přirovnává se k počtu hvězd ve vesmíru. Jednotlivé buňky jsou mezi sebou složitě propojeny a pospojovány do různých systémů. Celková délka nervových spojení u člověka dosahuje 150 000 km.



Obr.2.1. Znázornění stavby neuronu a jeho propojení s ostatními neurony [4]

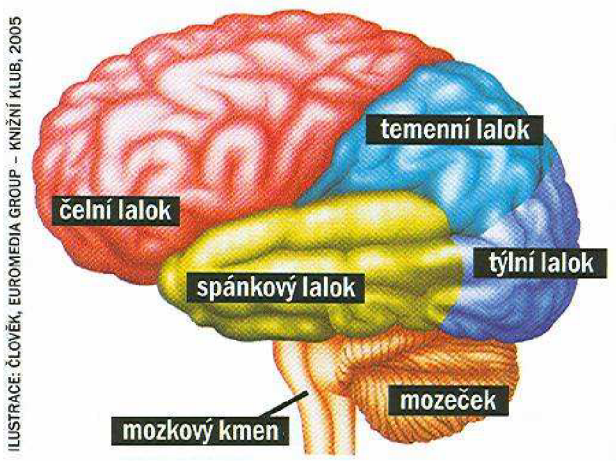
Neuron se primárně skládá z těla nervové buňky (soma), axonu a dendritů. Komunikace neuronů je založena na předávání drobných elektrických impulsů. Axon vede odstředivé (od těla buňky); dendrity vedou dostředivé. Vlastní předávání probíhá v tzv. synaptické štěrbině, která leží mezi knoflíkovitým zakončením axonu (synapsí) a dendritem druhého „přijímajícího“ neuronu. V synapsi se elektrický impuls buď přeměňuje na chemický pomocí transmiterů jako jsou acetylcholin, noradrenalin, dopamin, serotonin a další (v případě chemické synapse) nebo elektrický signál postupuje dále prostřednictvím těsných spojů mezi presynaptickým a postsynaptickým zakončením (elektrická synapse). Zde se uplatní kladné a záporné ionty nesoucí elektrický náboj.



Obr.2.2. Synaptický přenos [5]

Na mozek můžeme pohlížet jako na spojení několika částí s různým vývojovým stářím. Nejstaršími částmi jsou mozeček a mozkový kmen (souhrnný pojem pro prodlouženou míchu, most a střední mozek). Zajišťují v těle základní životní funkce, jako je dýchání řízení krevního oběhu a srdeční činnosti. Následuje mezimozek a limbický systém, který ovlivňuje naše chování a city. Vývojově nejmladší je koncový mozek složený z šedé kůry mozkové. Šedá kůra mozková se skládá ze 2 polokoulí a každá z nich obsahuje 4 laloky (čelní, temenní, týlní, spánkový), jejich názvy jsou odvozeny od názvů lebečních kostí, které je chrání. Zodpovídají za tyto funkce:

- čelní lalok - řeč, myšlení a pohyb končetinami
- spánkový lalok - sluch, paměť, rozumění řeči
- temenní lalok - vnímání dotyků a orientaci v prostoru
- týlní lalok - zrakové vnímání a poznávání předmětů



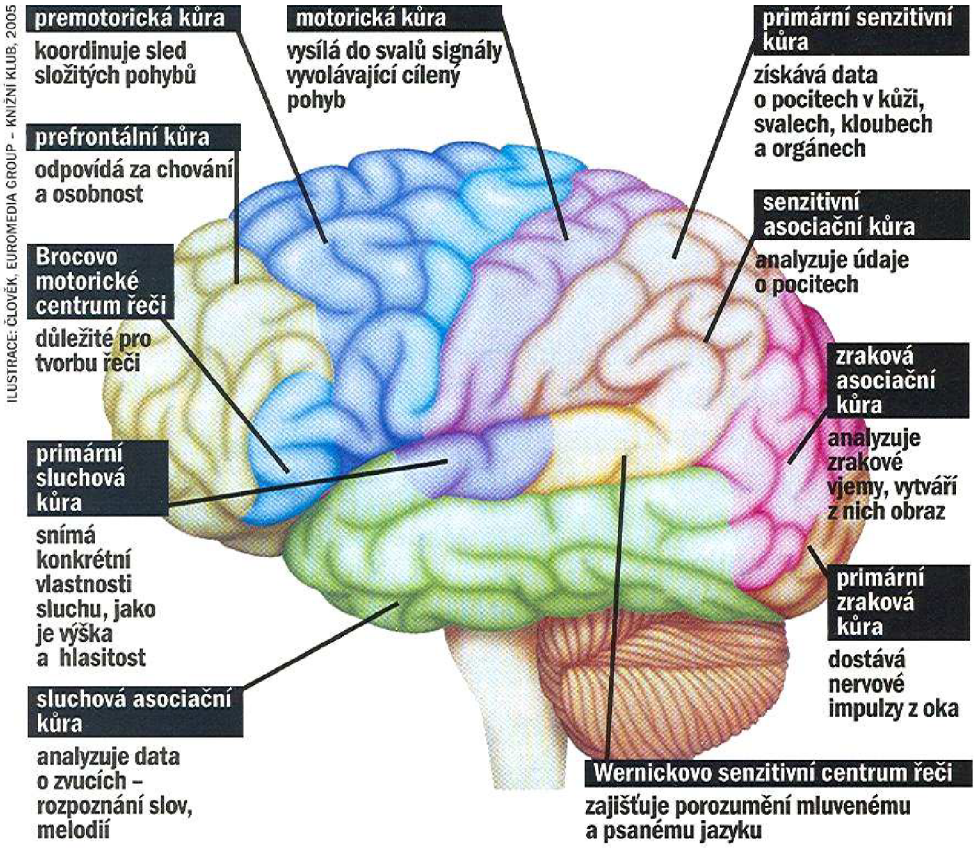
Obr.2.3. Rozmístění mozkových laloků [3]

Vlivem překřížení nervových drah vzniká tzv. obrácená laterita mozku:

- levá část mozku (levá hemisféra) má na starosti pravou polovinu těla
- pravá část mozku (pravá hemisféra) kontroluje levou polovinu těla

Psychické funkce jsou taktéž rozděleny. V levé části mozku, která je nejčastěji dominantní (většina lidí jsou praváci) má na starosti řeč, čtení, počítání. Nedominantní část řídí zase mimiku, poznávání, orientaci v prostoru atd.

Výzkumem šedé kůry mozkové se podařilo sestavit detailnější mapu jednotlivých oblastí odpovědných za vyšší mentální funkce (viz obr.2.4).



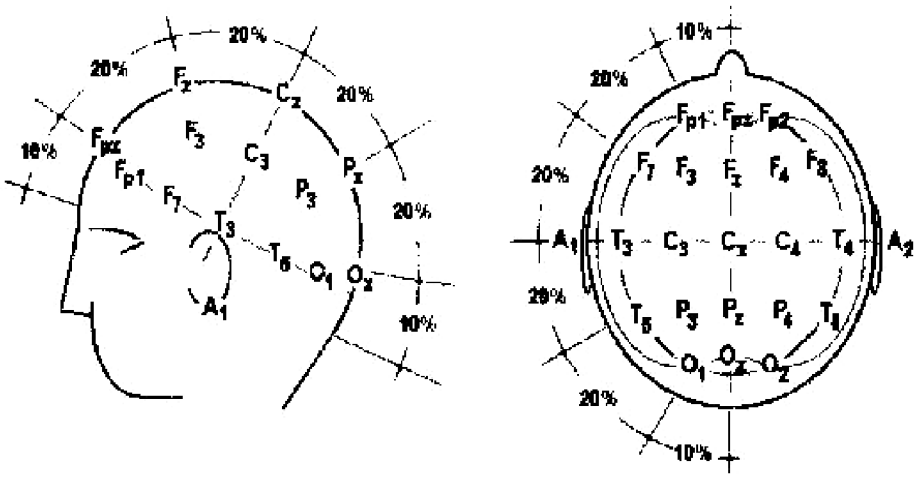
Obr.2.4. Mapa oblastí mozkové kůry dle funkce [3]

3. Základy elektroencefalografie (EEG)

Rytmická aktivace a inhibice skupin neuronů v mozku vysílá elektrické vlny, které se z nitra mozku dostávají na jeho povrch a posléze na pokožku hlavy. Poprvé se tímto jevem u člověka zabýval německý psychiatr Hans Berger (1873–1941). Zjistil, že po připojení kovových elektrod na povrch hlavy je možné zachytit citlivým galvanometrem elektrický proud. Položil tím základ nové lékařské diagnostické metodě – elektoencefalografii. Ta se postupem času stala jednou z hlavních metod při různých mozkových poruch, jako je například epilepsie a další.

Principiálně je zdrojem těchto vln každá nervová buňka, avšak přímé měření elektrického potenciálu jednotlivých buněk nebo malých částí neuronových polí je neuskutečnitelné. Prakticky měření probíhá rozdělením skalpu s oblastmi s elektrickou aktivitou na větší části odpovídající lalokům mozkové kůry. Tím získáme částečně zprůměrňované hodnoty pro danou oblast ležící pod elektrodou, protože jmenovité výkony jednotlivých seskupení neuronů s ohledem na jejich konkrétní funkce i lokalizaci se vždy poněkud liší. Pomocí EEG tedy měříme proměnlivé elektrické signály, jejichž zdrojem jsou synchronizované výboje nervových buněk, šířící se neuronovými okruhy.

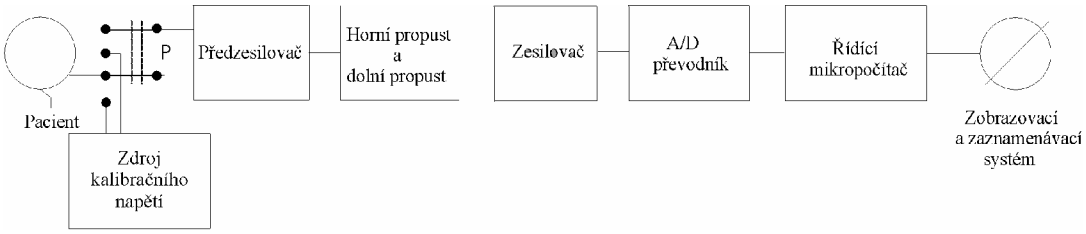
Mezinárodní systém 10–20 [5] z roku 1957 je nejrozšířenější standardem pro záznam EEG popisujícím rozmístění elektrod na skalpu. Systém je pojmenován podle poměrů vzdáleností mezi elektrodami vzhledem k vzdálenostem krajních elektrod. Pozice elektrod jsou na průsečících tohoto dělení. Každé místo je označeno písmenem (podle laloku kůry) a indexem tvořeným číslem nebo dalším písmenem (určuje hemisféru).



Obr.3.1. Schéma rozmístění elektrod systému 10–20 [4]

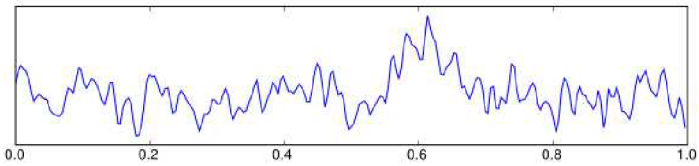
Použitá písmena znamenají: "F" – frontální (čelní) lalok, "T" – temporální (spánkový) lalok , "C" – centrální lalok (imaginární), "P" – prarientální (temenní) lalok, "O" – okcipitální (týlní) lalok. Sudá čísla indexů (2, 4, 6, 8) přísluší pravé hemisféře a lichá čísla (1, 3, 5, 7) levé hemisféře. "Z" označuje elektrody uprostřed hemisfér. S nižším číslem indexu se daná elektroda více blíží středu.

Elektrody jsou připojeny ke snímacímu zařízení – elektroencefalografu, který zajišťuje záznam signálů. Nejpoužívanější typy mají 16 až 32 kanálů, kdy každému kanálu přísluší několik elektrod (nejčastěji 3). Podle druhu záznamového média se dělí na analogové (papír) nebo digitální (elektronická paměť).



Obr.3.2. Blokové schéma jednoho kanálu digitálního elektroencefalografu [6]

Ze vstupních elektrod nebo ze zdroje kalibračního napětí je signál přes přepínač P veden kabelem na vstup předzesilovače, následují horno- a dolnopropustné filtry a zesilovač. Z něho je poté signál prostřednictvím A/D (analogově-digitálního) převodníku navzorkován do zdigitalizované podoby a dále zpracován digitální částí zařízení nebo počítačem. Mezi posledními prvky se může ještě objevovat řídící mikropočítač, který tvoří komunikační mezičlánek mezi těmito prvky. Ten zde může figurovat jako samostatná část nebo může být součástí zobrazovacího systému.



Obr.3.3. Příklad záznamu 1 sekundy signálu EEG (vzorkovací kmitočet 256Hz, elektroda umístěna na C_z, autorem tohoto a dále uvedených grafů se záznamem signálu je Hugo Gamboa) [7]

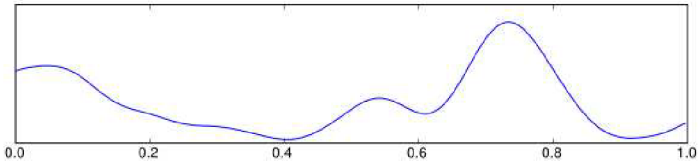
Typické parametry EEG zařízení [6]:

- Velikost vstupního napětí: 10–100 µV
- Citlivost přístroje 20–1000 µV
- Kmitočtový rozsah: 0,26–100 Hz
- Vstupní odpor: 1–100MΩ

Rozdělení vln EEG podle kmitočtu [7] [8]:

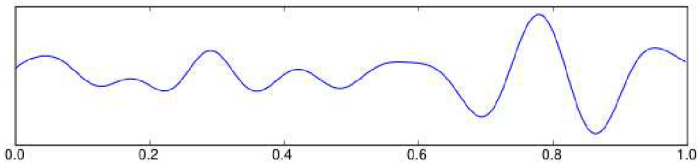
Delta

Vlny Delta se nalézají v pásmu od 3 Hz dolů. Mají sklon mít nejvyšší amplitudu a nejnižší kmitočet. Převládají u dětí do jednoho roku věku a v hlubokém spánku. Mohou být vyvolány lokálními podkorovými lézemi a obecně šířením z rozptýlených lézí. Obvykle jsou nejvíce patrné v předních částech mozku u dospělých a v zadních u dětí.



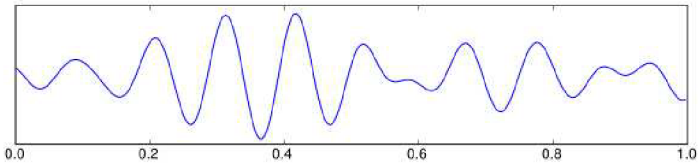
Obr.3.4. Vlny delta [7]

Theta
Vlny Theta jsou v rozmezí 3,5 až 7,5 Hz a považují se za „pomalou" aktivitu mozku. Jsou abnormální u bdějících dospělých, ale normální u dětí do 13 let a ve spánku. Zdroje tohoto vlnění mohou být spjaty s lokálními lézemi v podkorové oblasti.



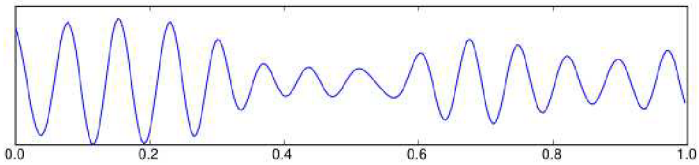
Obr.3.5. Vlny theta [7]

Alfa
Vlny Alfa mají kmitočet mezi 7,5 a 12 Hz. Vyskytují se především v zadních částech hlavy po obou stranách s vyšší amplitudou v dominantní hemisféře. Jsou vyvolány zavřením očí a uvolněním; mizí s otevřením očí nebo zásahem vyšších mentálních funkcí (myšlení, počítání). Převládají v normálním uvolněném stavu u dospělých, vyskytují se jinak po celý život, obzvláště kolem třináctého roku věku.



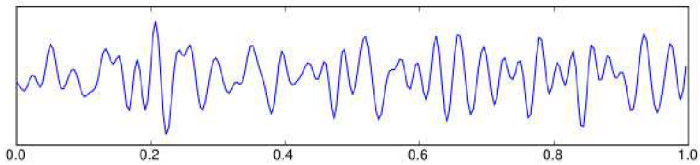
Obr.3.6 Vlny alfa [7]

Senzomotorický rytmus (SMR)
Jsou to „středně rychlé" vlny v rozsahu zhruba 12–16 Hz. Tyto vlny vznikají senzomotorické kůře, odpovědné za třídění vizuálních podnětů a koordinaci svalových pohybů. Pojí se s psychickou stabilitou a koncentrací na fyzickou aktivitu.



Obr.3.7 Vlny SMR [7]

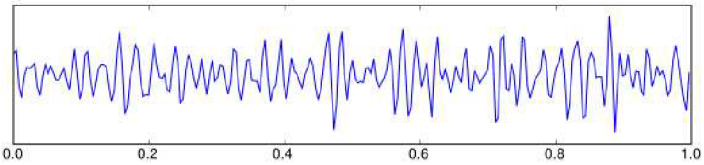
Beta
Vlny Beta se též nazývají „rychlou" aktivitou mozku. Jejich kmitočet je od 16 Hz výše. Obvykle se vyskytují souměrně po obou stranách hlavy s nejvyšším výskytem v její přední části. Mohou být potlačeny nebo zcela chybět v místech poškození mozkové kůry. Jde o normální typ mozkových vln. Převládá u pacientů s úzkostí a napětím, nebo při zavření očí.



Obr.3.8. Vlny beta [7]

Gama

Vlny Gama produkují skupiny neuronů při vrcholném vybuzení spojeném s hyperpsychologickým stavem. Většinou vznikají při úzkosti, hyperaktivitě, tenzi a těžké tělesné námaze. Můžeme je nalézt u schizofreniků a osob trpících rozštěpením osobnosti.



Obr.3.9. Vlny gama [7]

4. Principy EEG biofeedbacku

Termín biofeedback je užíván ve spojitosti s biologickou zpětnou vazbou. Biologická zpětná vazba je založena na audiovizualizaci některých psychosomatických veličin a procesů člověka s cílem jejich autoregulace. Můžeme měřit tepovovou frekvenci, krevní tlak, bioelektrický výkon mozkových center nebo elektrický odpor pokožky. Funkci regulátora při tom plní terapeut, většinou se jedná o automatizovaný systém typu vestavěného zařízení nebo osobního počítače.

Hlavní typy biofeedbacku podle měřené veličiny [9]:

Elektromyografie (EMG)
Poměrně rozšířený způsob, kdy se elektrodami či jinými senzory měří hodnoty svalového napětí (tonusu). Takto koncipovaný systém (viz obr. 4.1.) dokáže uživatele upozornit na výskyt svalového napětí a prostřednictvím zpětné vazby mu umožňuje toto napětí regulovat. Hlavní využití této metody se nachází v oblasti technik relaxace svalů při bolestech zad, hlavy, krku a syndromu skřípání zubů (tzv. bruxismus).

Povrchová tělní teplota
Čidla pro měření tělní teploty jsou umístěna na prstech rukou nebo na chodidlech. Protože tělní teplota často klesá se zvýšeným stresem, je možné cílenou regulací teploty dosáhnout uvolnění celého těla.

Galvanický odpor kůže
Pomocí senzorů na povrchu kůže se prostřednictvím galvanického odporu měří aktivita potních žlázek a množství jimi vyprodukovaného potu. Takto získaná informace slouží k léčbě fobií, úzkosti a koktání. Nejznámější příkladem použití je detektor lži. Velkou popularitu si získaly hry určené pro širokou veřejnost, jako je například Journey to Wild Divine (*www.wilddivine.com*).

Elektroencefalografie (EEG)
Využívá EEG zařízení ke snímání bioelektrické aktivity mozku spojené s různými stavy lidského vědomí jako je bdělost, uvolněnost, klidový stav, lehký a hluboký spánek. Tato metoda dnes patří mezi nejméně používané, hlavně z důvodu vysoké ceny a špatné dostupnosti vhodných zařízení EEG. Je však možné, že v budoucnu s příchodem jednoduchých a levných zařízení se tato situace změní, zvláště pokud bude dané zařízení víceúčelové (viz kapitola 8).

Obr.4.1. Příklad systému pracující s bioelektrickou aktivitou svalů (elektromyografie) [10]

V 60. letech 20. stol. byla profesorem Stermanem vyvinuta terapeutická metoda, která je založena na získávání obrazu o aktivitě lidského mozku a tím nepřímo i informaci o tělesném a duševní stavu člověka. Jejím základem je aplikace poznatků o vlnách EEG a vlastnostech jednotlivých kmitočtů. Praktické rozdělení vln mozkových vln a jejich vliv na psychiku člověka [11]:

Frekvenční pásmo:		**Mentální stav:**
Delta	1 – 3 Hz	hluboký spánek
Theta	4 – 7 Hz	povrchní spánek, útlum, denní snění, meditace, hypnóza
Alfa	8 – 12 Hz	základní bdělost (zavřené oči, nicnedělání)
SMR	12 – 15 Hz	uvolněná pozornost, autoregulace
Beta	15 – 20 Hz	soustředěná pozornost, volní koncentrace
Beta 2	21 – 30 Hz	hyperexcitace - napětí, podráždění, úzkost
Gama	31 a více Hz	špičkové výkony, vrcholné prožitky

V současné době existuje na trhu několik typů přístrojů, které pracují na základě audiovizuální stimulace (psychowalkmany) nebo snímaní biopotenciálů (EMG biofeedback, EEG biofeedback). Prostřednictvím biofeedbacku můžeme ovlivnit celou řadu vlastností mozku, jako je schopnost relaxace, zvládání stresových situací nebo zlepšení výkonu intelektu. Vynikající a rychlý účinek má také na poruchy spánku, zejména problémy s usínáním.

Obr.4.2. Psychowalkman [12]

EEG biofeedback

EEG biofeedback je ze všech typů biofeedbacků nejpřímočařejším a pravděpodobně i nejúčinnějším řešením. Pracuje totiž přímo s jednotlivými pásmy vlnění snímaného EEG zařízením a umožňuje člověku přímo ovládat své mozkové vlny učením mozku pomocí biologické zpětné vazby.

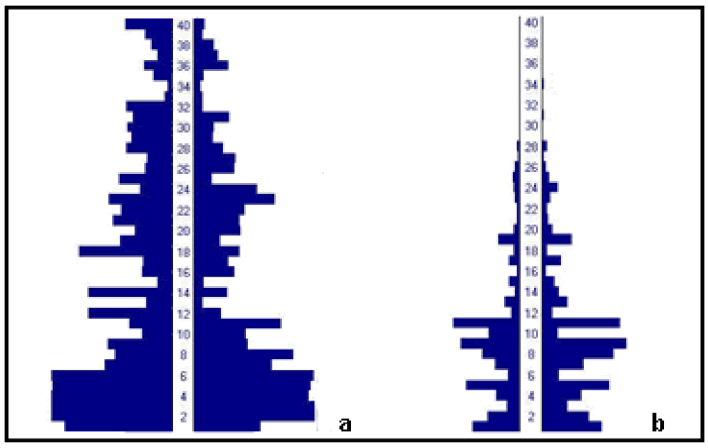
Zevrubně lze tuto metodu popsat následovně:

- elektrody přilepené k hlavě snímají mozkové vlny, které se zesilují a digitalizují EEG zařízením a následně sledují v počítači
- počítač zpracovává signály a zajišťuje zpětnou vazbu, což je informace o tom, jak mozkové vlny odpovídají nastavení terapeuta v konkrétním okamžiku
- po vyhodnocení zpětné vazby se výsledek transformuje do audiovizuální podoby prostřednictvím počítačové hry, kde je jediným ovládacím prvkem lidská mysl
- s narůstající aktivitou v preferovaných pásmech a poklesem v nežádoucích roste i úspěšnost hráče, v opačném případě dochází k poklesu úspěšnosti

Mozek postupně reaguje na motivační vodítka, které mu počítač poskytne tím, že ho odměňuje za dobré výsledky ve hře. Tak mozek sám rozvíjí proces učení nových, vhodnějších frekvencí mozkových vln.

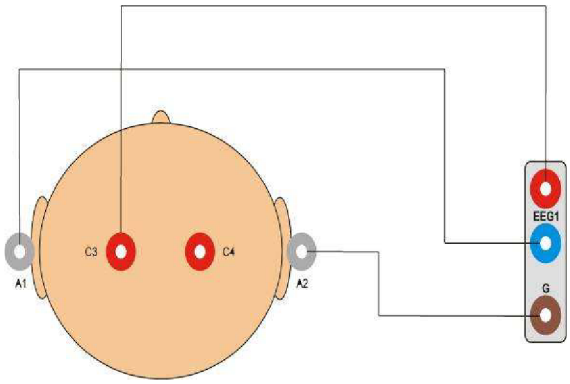
Obr.4.3. Ilustrační obrázek EEG biofeedbacku, kde je vidět v pozadí počítač se spuštěnou hrou, v popředí počítač terapeuta [13]

Správné rozložení aktivity ve spektru EEG signálu má rozhodující význam pro správnou funkci lidského mozku a celé nervové soustavy [13]. Na obr. 4.4. je zcela jasně vidět rozdíl mezi výrazně abnormálním (*a*) a normálním spektrem (*b*). Zatímco ve spektru *a* lze pozorovat vysokou úroveň aktivity na počátku a konci spektra s výrazným úbytkem mezi 12–15 Hz, tak ve spektru *b* je dominantní aktivita v pásmu alfa s méně výraznými vrcholy v ostatních pásmech a je patrná nízká aktivita nad kmitočtem 23 Hz s vyvážeností mezi oběma hemisférami. Špatná souhra mozkových vln v případě *a* u pacienta způsobila roztěkanost, impulsivitu, emoční labilitu a poruchu spánku.



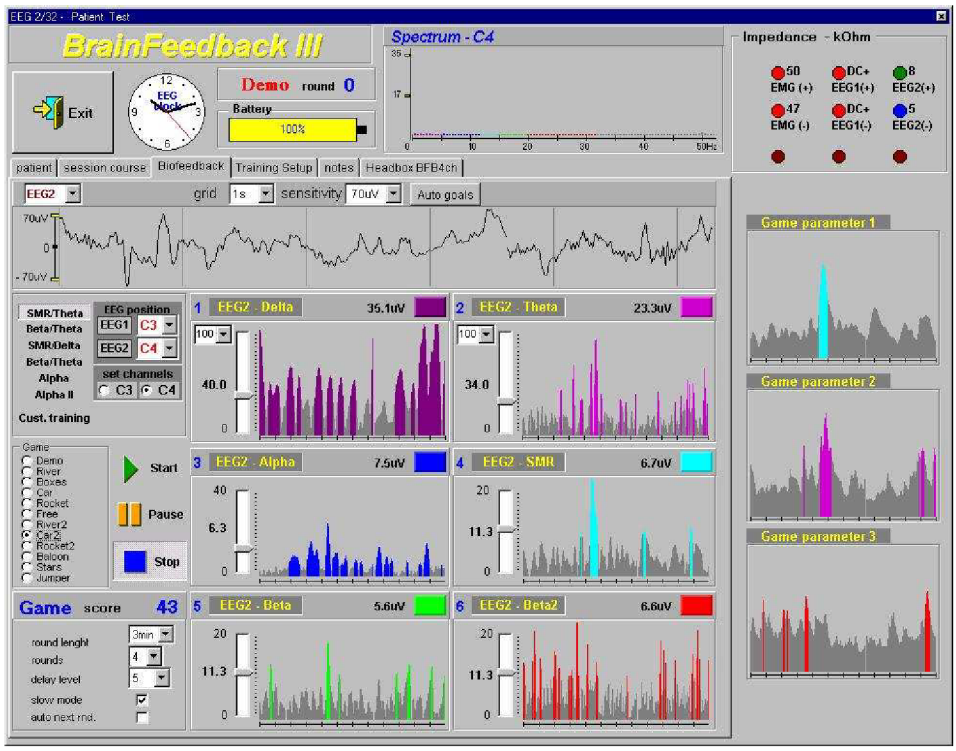
Obr.4.4. Výrazně abnormální (a) a přiměřené (b) spektrum signálu EEG [14]

Nejčastěji se pro snímání vln EEG používá pouze jeden kanál. Je to dáno tím, že se trénuje jenom jedna z hemisfér (viz obr. 4.5.). V závislosti na potížích pacienta to může být levá hemisféra (měření se provádí elektrodou C3) nebo pravá (měříme na C4). Druhá elektroda kanálu je připojena na ušní lalůček na dané straně. Nulová elektroda je pak připojena k uchu na straně opačné.



Obr.4.5. Příklad zapojení pro trénink levé hemisféry [15]

Nastavení reakcí hry na hodnoty jednotlivých pásem EEG je pro úspěch této metody klíčové [13]. Tradiční je takzvaný Beta–SMR tréning, kdy jsou posilovány pásma Beta a SMR. Příklad uživatelského rozhraní pro nastavování parametrů systému je možno vidět na obr.4.6., uprostřed okna jsou posuvníky pro nastavení úrovní jednotlivých pásem.



Obr.4.6. Ovládací panel terapeuta v komerčním systému [15]

5. Specifikace systému pro EEG biofeedback

Cíle projektu

Hlavním cílem projektu je návrh a implementace software s volně dostupnými zdrojovými kódy (např. s licencí GPL) umožňující široké skupině lidí trénovat svůj mozek pomocí biologické zpětné vazby na běžném osobním počítači. Výsledkem by měla být jednoduše ovladatelná aplikace, kterou by po určitém seznámení s problematikou mohli používat i úplní laici bez lékařského nebo technického vzdělání. Pro co nejširší uživatelskou základnu by měl být systém snadno přenositelný i na jiné operační systémy nebo na jinou platformu, než jak bylo původně navrženo a posléze implementováno.

Příslušný software je třeba doplnit o vhodný hardware. Zde se přímo nabízí použití open–source zařízení ModularEEG, vyvíjeného v rámci mezinárodního projektu OpenEEG (http://openeeg.sourceforge.net/doc/). Taková kombinace dohromady utvoří kompletní celek, který není závislý na komerčních společnostech a jejich proprietárních technologiích a postupech. Tímto se ovšem samozřejmě nevylučuje a nijak „nezakazuje“ implementace podpory pro zařízení vyvíjená na komerční bázi.

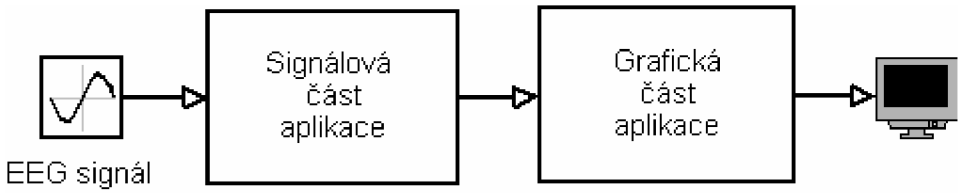
Požadavky kladené na vyvíjenou aplikaci:

- implementace na platformě IA-32 (x86) s operačním systémem MS Windows, implementace se bude postupně dále rozšiřovat na další platformy a operační systémy
- pohodlný instalátor bez nutnosti instalovat ručně podpůrné knihovny či jiné části aplikace
- plná podpora zařízení ModularEEG s výhledem na další rozrůstání množiny podporovaných zařízení
- automatická detekce portu, na kterém je zařízení připojeno (užitečné, pokud používáme převodník USB↔RS232, u kterého si ovladač náhodně vybírá číslo portu…)
- volba snímaných kanálů nezávisle na snímaných pásmech
- možnost flexibilně upravovat hraniční kmitočty jednotlivých pásem
- celoobrazovkový grafický výstup s volbou rozlišení a obnovovací frekvence obrazu

6. Návrh programového vybavení

Objektově orientované programování (OOP) patří v současné době k nejrozšířenějším metodologiím návrhu a implementace software, pokud neuvažujeme o oblasti systémového programování a operačních systémů. Jak se OOP postupně vyvíjelo a složitost navrhovaných systémů narůstala, vznikla určitá množina doporučení o tom, jaké vztahy mezi třídami použít, jak by se dané třídy a jimi vytvářené objekty měly chovat a jak vytvářet sady objektů specifické pro určitou situaci. Tento katalog doporučení je znám pod pojmem návrhové vzory (design patterns) [16]. Zmíněné vzory se často opakují a jsou používány v mnoha programovacích jazycích, proto jsem se jimi snažil inspirovat a vytvořit objektový model, který bude splňovat kritéria kvalitního návrhu.

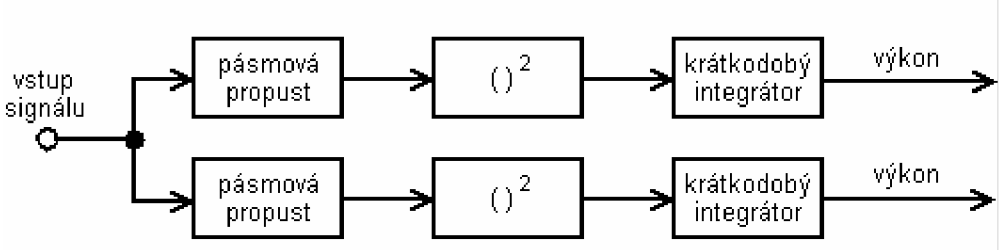
K vlastní prezentaci návrhu je použit jazyk UML (Unified Modeling Language) [17], konkrétně digramy tříd a sekvenční diagramy, obojí doplněné textovým popisem. Protože je problematika zpracování signálu odlišná od programování grafických aplikací a her, rozdělil jsem popis návrhu do dvou samostatných podkapitol. Obě části na sebe navazují, jak je možno vidět na obr. 6.1.



Obr.6.1. Rozdělení návrhu aplikace

6.1 Návrh signálové části

Z principu EEG bifeedbacku uvedeném v kapitole č.4 vyplývá, že hlavním úkolem signálové části aplikace by mělo být čtení dat ze vstupního zařízení a analýza úrovně signálu ve sledovaných pásmech. Čtení dat je spojeno s implementací komunikačního protokolu a bude tedy uvedeno v následující kapitole, zabývající se implementací. Získávání síly signálu v určitém pásmu je možno realizovat 2 způsoby. První způsob spočívá v užití rychlé *Fourierovy transformace* (algoritmus FFT– Fast Frourier Transform) z časové oblasti do kmitočtového spektra a integraci hodnot ve spektru pro požadovaná pásma. Druhý způsob je postaven na základě upravené *metody pásmové lineární filtrace* [18], která spočívá ve vytvoření banky pásmových propustí, kdy je výstup každé propusti umocněn na druhou a zpracován krátkodobým integrátorem (viz obr.6.2.). Na rozdíl od původní metody pásma v tomto případě na sebe nenavazují, ale představují konkrétní kmitočty mozkových vln, které chceme sledovat.



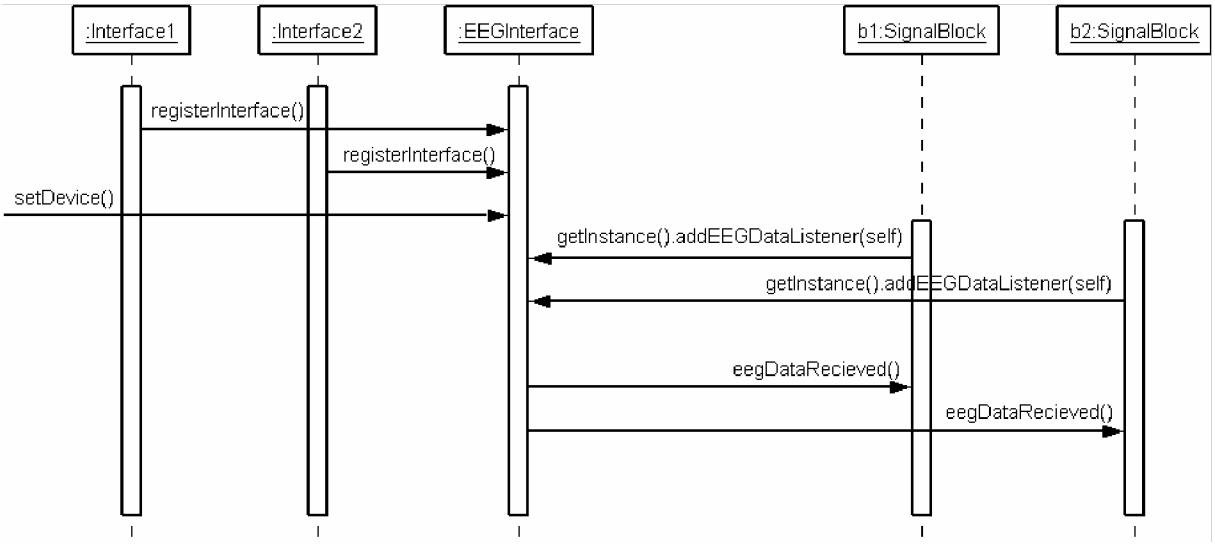
Obr.6.2. Příklad sledování 2 pásem bankou pásmových propustí [18]

K nevýhodám *krátkodobé Fourierovy transformace* patří především zvýšená náročnost na výpočetní výkon počítače. Dále má omezenou rozlišovací schopnost událostí v čase danou velikostí použitého okna (počtu zpracovávaných vzorků). Tyto nedostatky zcela odstraňuje druhá metoda, kdy potřebujeme pouze filtry nutné k sestavení pásmových propustí a s každým novým vzorkem na vstupu obdržíme aktuální hodnotu na výstupu. Uvedené skutečnosti mě přiměly k tomu, abych zvolil druhou metodu.

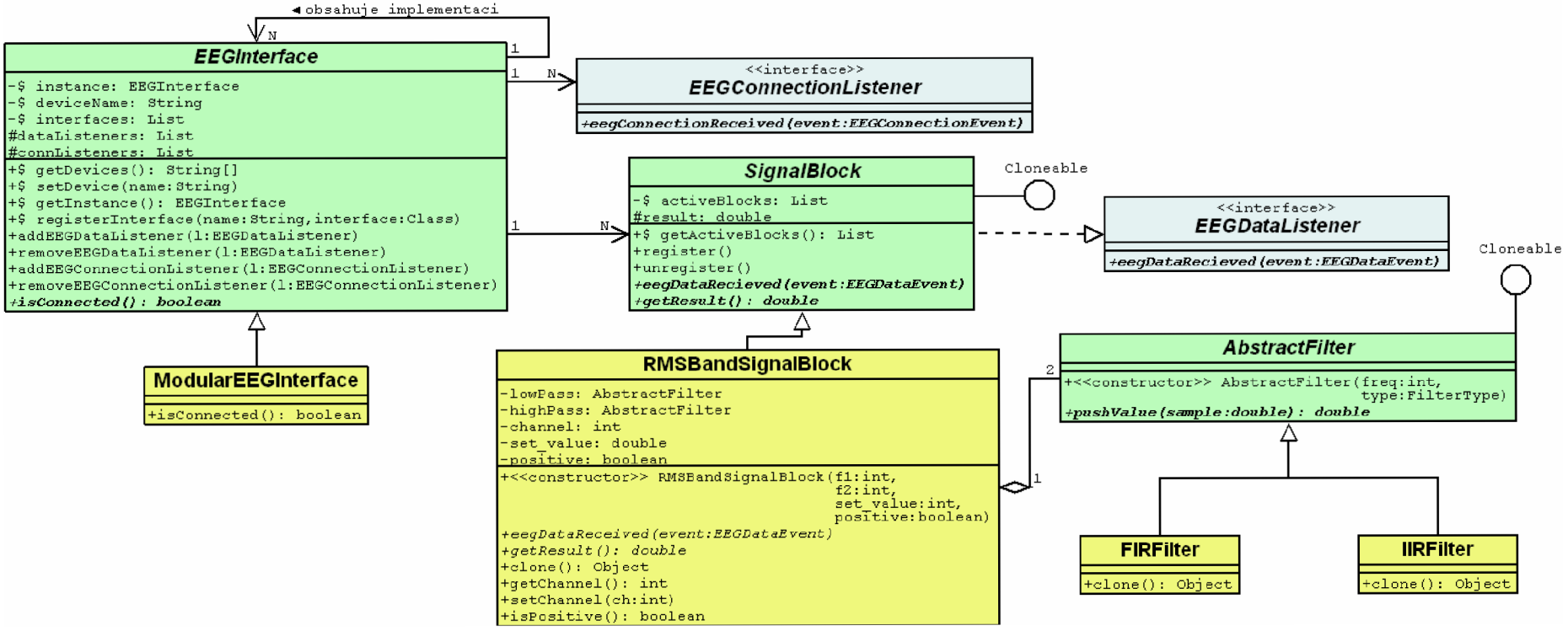
Algoritmy výpočtu koeficientů pásmových propustí nejsou příliš triviální, proto využijeme specializované programy pro návrh filtrů (např. Matlab), které umí vypočítat hodnoty koeficientů a zapsat je do binárního souboru. Pro realizaci pásmových propustí je výhodné použít kombinaci doplní a horní propusti, která dohromady realizuje propust pásmovou, protože jinak bychom byli nuceni generovat všechny možné kombinace hraničních kmitočtů. Pro rozsah frekvencí 1 až 50 Hz při kroku 1 Hz a podmínce, že horní kmitočet propusti je vyšší než dolní, by to znamenalo výpočet koeficientů 1225 pásmových propustí! Ve druhém případě potřebujeme pouze 50 dolních propustí a 50 horních propustí. Pro jejich realizaci lze použit buď filtry s nekonečnou impulsní odezvou (typ IIR–Infinite Impulse Response) nebo s konečnou impulsní odezvou (FIR–Finite Impulse Response) .

Nyní se dostáváme k samotné analýze entit potřebných v signálové části. Zcela určitě budeme potřebovat třídu abstraktního rozhraní, které bude zapouzdřovat jednotlivé komunikační protokoly specifického hardware a bude poskytovat snímaná data a informace o stavu konkrétního zařízení. Pro realizaci schématu na obr.6.2 je navíc třeba do návrhu umístit třídu objektů, které budou tyto data číst, zpracovávat a na základě tohoto zpracování poskytovat jednu výstupní hodnotu odpovídající vstupním datům. Jelikož budeme pásmové propusti sestavovat „uživateli na míru“ až za běhu programu, je nutné také navrhnout třídy reprezentující filtry IIR a FIR, které jsou schopny podle zadaného kmitočtu si samy nahrát koeficienty z binárních souborů. Na základě těchto znalostí jsem navrhl digram tříd, jehož podobu lze vidět na obr.6.4.

Chování tříd a objektů signálové části s důrazem na samotný přístup ke třídě *EEGInterface* (třída zapouzdřující jednotlivá rozhraní, viz dále) znázorňuje sekvenční diagram na obr.6.3.



Obr.6.3. Sekvenční diagram



Obr.6.4. Diagram tříd signálové části

Popis entit v diagramu tříd:

EEGInterface
Abstraktní třída vstupního rozhraní použitá jako základ pro implementaci různých komunikačních protokolů konkrétních zařízení EEG. Jednotlivé neabstraktní podtřídy se automaticky registrují při startu aplikace voláním statické metody *registerInterface()* svého předka, kdy poskytnou svůj identifikační řetězec (např. jméno zařízení) a referenci na svou třídu. Uživatel si potom může vybrat z nabídky všech podporovaných zařízení v systému, jejichž identifikační řetězce vrací statická metoda *getDevices()*, a nastavit dané rozhraní statickou metodou *setDevice()*. Protože rozhraní, ze kterého se čtou data, může být v systému v daný moment pouze jednou, je třída *EEGInterface* navržena pomocí vzoru Jedináček (Singleton) a vytvoří pouze jednu svou instanci dostupnou při volání statické metody *getInstance()*. Dalším voláním metody *setDevice()* se vytvořená instance zahodí a vytvoří nová, která odpovídá novému rozhraní, aby měl uživatel možnost měnit vstupní zařízení bez restartu aplikace. S ostatními objekty signálové části *EEGInterface* komunikuje prostřednictvím zasílání zpráv *ConnectionEvent* a *EEGDataEvent*. Zpráva *ConnectionEvent* je zasílána registrovaným příjemcům při připojení nebo odpojení vstupního zařízení a informuje o jeho stavu, včetně použitého portu počítače. Druhá zpráva *EEGDataEvent* slouží k posílání dekódovaných dat různým zpracovatelům signálů.

ModularEEGInterface
Implementace rozhraní pro čtení dat z ModularEEG. Registruje se u svého předka *EEGInterface* jako „ModularEEG“. Další popis této třídy obsahuje kapitola o implementaci.

EEGDataListener
Rozhraní pro objekty přijímající data ze vstupního rozhraní.

EEGConnectionListener
Rozhraní příjemců zprávy o stavu zařízení.

SignalBlock
Abstraktní třída bloku zpracovávajícího blíže nespecifikovaným způsobem vstupní data z EEG zařízení a poskytující na výstupu přístupném přes metodu *getResult()* výsledek typu double. Konkrétní funkce bloku je implementována v podtřídách třídy *SignalBlock*, které volají její metodu *register()* v momentě, kdy mají správně nastaveny všechny parametry potřebné pro zpracování signálu a jsou schopny přijímat data. Rodičovská třída *SignalBlock* je pak zařadí do seznamu aktivních bloků přístupného přes metodu *getActiveBlocks()* navazujícím částem aplikace. Pokud již daný blok není potřeba, je zavolána metoda *unregister()*. Aby bylo možné bloky, které jsou si vzájemně podobné, jednoduše kopírovat podle návrhového vzoru Prototyp (Prototype), je implementováno rozhraní *Cloneable*.

RMSBandSignalBlock
Objekty této třídy odvozené od třídy SignalBlock slouží určování poměru úrovně síly signálu v pásmu vymezeném kmitočty *f1* a *f2* vůči hodnotě nastavené hodnotě *set_value*. Realizovaná funkce je následující:

$$result(x_1..x_N) = \frac{\frac{\sum x_i^2}{N}}{set\_value}$$

Metoda *setChannel()* slouží k nastavení čísla zpracovávaného kanálu EEG zařízení (pokud má více kanálů) a metoda *isPositive()* informuje navazující části, zda rostoucí výstupní hodnota daného bloku je pozitivní (síla signálu v tomto pásmu se má zvyšovat) nebo negativní (síla signálu se má snižovat).

AbstractFilter
Abstraktní třída *AbstractFilter* představuje obecný model filtru, kdy pomocí metody *pushValue()* na vstup filtru vložíme vzorek signálu a zároveň dostaneme přes návratovou hodnotu výstupní hodnotu. Parametr konstruktoru *freq* určuje mezní kmitočet filtru a parametr *type* jeho typ (např. dolní nebo horní propust). Z důvodu podpory návrhového vzoru Prototyp (Prototype) pro třídu *RMSBandSignalBlock* je implementováno rozhraní *Cloneable*.

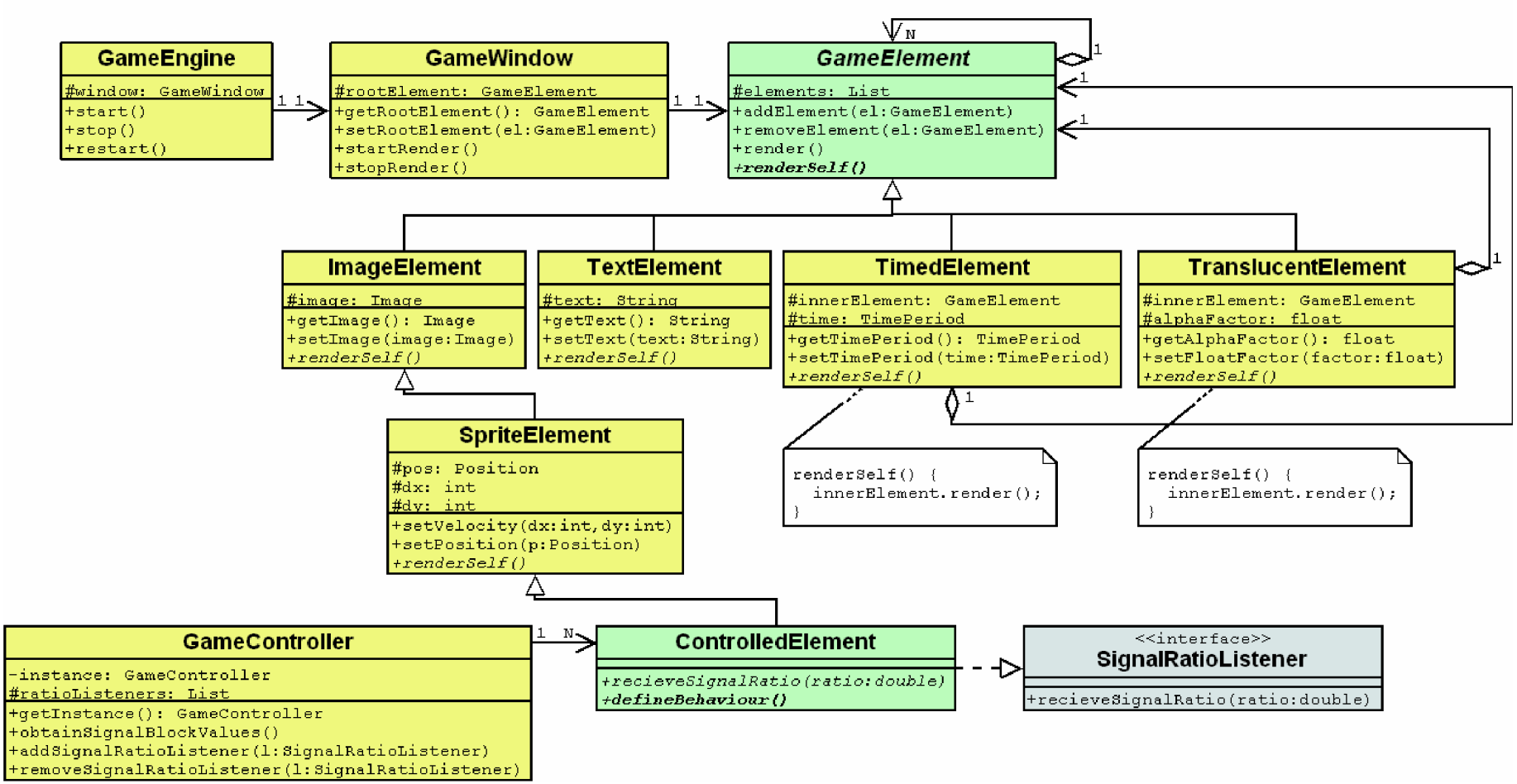
FIRFilter
Třída s implementací filtru s konečnou impulsní odezvou.

IIRFilter
Třída s implementací filtru s nekonečnou impulsní odezvou.

6.2 Návrh grafické části

Grafická část aplikace má za úkol určitým způsobem zobrazovat uživatelovy úspěchy při vylaďování bioelektrické aktivity mozku a motivovat jej k lepším výsledkům. K tomu účelu se většinou používá počítačová hra, ve které hráč nepřímo ovládá rychlost a/nebo směr pohybu nějakého objektu pohybujícího se po obrazovce. Zobrazovaná scéna může být koncipována jako trojrozměrná (3D) nebo dvojrozměrná (2D) se statickým nebo pohyblivým pozadím. Pro přímočarost návrhu grafické části a jednoduchost výsledné aplikace jsem zvolil variantu dvojrozměrné grafiky se statickým pozadím.

Začněme tedy analýzu potřebných tříd a objektů. Jako hlavní objekt potřebujeme tvůrce scény, který ji vytvoří a průběžně aktualizuje po celou dobu běhu programu. Návrh musí také obsahovat renderovací smyčku, která na obrazovku průběžně vykresluje samotnou scénu. Tuto scénu je vhodné rozdělit na prvky (elementy), které reprezentují její jednotlivé vykreslované části (např. obrázky, text). Elementy by bylo možné do sebe vnořovat, přičemž každý element by byl zodpovědný za vykreslení elementů v něm obsažených. Zvláštním případem jsou elementy řízené signálovou částí. U každého z nich by mělo být možné explicitně definovat jeho chování. Spojovacím článkem mezi signálovou a grafickou částí bude objekt sbírající výsledky všech aktivních signálových bloků a poskytující sjednocenou informaci těmto elementům. Navržený diagram tříd ilustruje obr. 6.5.



Obr.6.5. Diagram tříd grafické části

Popis entit v diagramu tříd:

GameEngine
Hlavní třída grafické části, která sestavuje zobrazovanou scénu a řídí její vykreslování podle dané časové sekvence specifikované v implementaci třídy. Metody *start()* a *stop()* slouží k zahájení a přerušení této sekvence. Po zavolání metody *restart()* se veškerý děj vrací na začátek. Ukončení je provedeno automaticky po určitém nastaveném čase.

GameWindow
Celoobrazovkové okno, do kterého se probíhá samotné vykreslování scény. Třída obsahuje referenci na kořen stromu elementů scény *rootElement*, které se jeho průchodem vykreslují. Metody *getRootElement()* a *setRootElement()* jsou určeny k získání respektive k nastavení tohoto kořene. Vlastní vykreslování probíhá ve smyčce vykonávané samostatným vláknem, které se vždy po zobrazení jednoho snímku na chvíli uspí, aby nebyl zbytečně zatěžován procesor. Vykreslovací vlákno lze pozastavit metodou *stopRender()* a znovu spustit metodou *startRender()*.

GameElement
Abstraktní třída prvku, ze kterých je složena grafická scéna. Byl použit návrhový vzor Skladba (Composite) [16] v upravené podobě, která neobsahuje explicitní uzlové třídy. Každý element může obsahovat vnitřní elementy odkazované seznamem *elements*. Je-li seznam prázdný, jedná se o uzlovou třídu s jedním elementem. Jednotlivé elementy se dají přidávat a ubírat z rodičovského elementu pomocí metod *addElement()* a *removeElement()*. Metoda *render()* slouží k vykreslení celé stromové struktury, zatímco metoda *renderSelf()* je ponechána jako abstraktní a provádí vykreslení konkrétního prvku.

ImageElement
Třída elementu obsahujícího obrázek. Metodami *getImage()* a *setImage()* lze přistupovat k samotnému obrázku.

TextElement
Třída elementu s textem. Metoda *getText()* vrací aktuálně zobrazovaný text a metoda *setText()* jej přepíše zadaným parametrem *text*.

TimedElement
TimedElement je obalová třída elementu, která k existující třídě odvozené od *GameElement* zajistí přidání nové vlastnosti. Tento princip odpovídá návrhovému vzoru Dekorátor (Decorator) [16]. V tomto případě jde o zobrazení prvku *innerElement* v časovém rozmezí definovaném proměnnou *time*, se kterou se pracuje prostřednictvím metod *getTimePeriod()* a *setTimePeriod()*.

TranslucentElement
Obalová třída elementu pro přidání průsvitnosti. Ta je specifikována proměnnou alphaFactor, která určí míru průsvitnosti elementu pomocí tohoto vztahu (technika Alpha Blending):

$$pix[x, y] = \alpha \cdot element[x, y] + (1 - \alpha) \cdot background[x, y]$$

SpriteElement
Pohyblivý element (sprajt) odvozený od třídy *ImageElement*, který má přiřazenu pozici na obrazovce a vektor rychlosti, se kterou se pohybuje. Rychlost je rozdělena na složky *dx* a *dy*, které určují přírůstky pro osy x a y. Vektor rychlosti se nastavuje přes metodu *setVelocity()* a metoda *setPosition()* umístí sprajt na požadovanou pozici.

GameController
Třída zajišťující sběr výsledků ze všech aktivních signálových bloků, jejich zpracování do jediné výsledné hodnoty a rozeslání výsledku zaregistrovaným příjemcům. Protože pouze jedna instance této třídy může v daný moment existovat, je použit návrhový vzor Jedináček (Singleton) a přístup k unikátní instanci třídy je možný pouze přes statickou metodu *getInstance()*. Metoda *obtainSignalBlockValues()* zajistí načtení výsledků ze všech aktivních signálových bloků. Metody *addSignalRatioListener()* a *removeSignalRatioListener()* pak slouží k přidání a odebrání příjemců výsledné hodnoty.

SignalRatioListener
Rozhraní příjemců zprávy o výsledku z třídy *GameController*.

ControlledElement
Abstraktní třída elementu, který je řízen výsledky biofeedbacku. Metoda *defineBehaviour()* specifikuje chování konkrétního elementu po obdržení vstupní hodnoty metodou *recieveSignalRatio()*.

7. Implementace systému

Jednou z klíčových otázek při implementaci každého programového systému je správná volba vhodného implementačního jazyka. Rozhodnutí je ovlivněno především stávajícími hardwarovými a softwarovými prostředky, na nichž má daný systém běžet nebo s nimiž má spolupracovat, dále pak zkušeností vývojářů s příslušnými technologiemi a ochotou vyzkoušet si nové postupy a platformy.
V mém případě se jedná o systém víceméně nezávislý na použitém operačním systému a hardwarovém vybavení. Jediným zásadním kritériem výběru byla schopnost grafického výstupu na obrazovku a možnost číst vstupní data sériovou linkou dle standardu RS-232. Uvedu zde několik důvodů, pro které byla implementačním jazykem zvolena Java:

- je to již léty prověřená a stabilní platforma, kterou používají takové instituce, jako je například americká NASA
- umožňuje přenositelnost mezi různými architekturami počítačů a operačními systémy bez nutnosti znovu překládat zdrojové kódy
- návrh jazyka obsahuje mechanismy, které snižující počet chyb a vedou programátora k přehlednějšímu a bezpečnějšímu kódu
- přestože rychlost aplikací napsaných v Javě je obecně nižší, než v jazycích kompilovaných do strojového kódu, tak s použitím technologie HotSpot se tyto rozdíly zmenšují
- snadnost implementace nových a rozšiřování stávajících systémů založených na objektech a objektově orientovaném programování

Svůj program jsem poněkud neoriginálně nazval Jfeedback, inspirován spojením slov biofeedback a Java.. Jelikož přepokládám, že čtenář je dostatečně obeznámen se základními technikami programování v Javě, budu v následujících podkapitolách popisovat pouze význačné problémy, se kterými jsem se při implementaci Jfeedbacku potýkal, či jinak zajímavé skutečnosti a podněty.

7.1 Implementace signálové části

Registrace potomků třídy EEGInterface

Třída *EEGInterface* je navržena podle návrhového vzoru Jedináček (Sigleton) [16] s tím, že bude odstiňovat komunikaci s konkrétním zařízením a poskytovat jednotné rozhraní pro ostatní třídy a objekty v programu. Jednotlivé implementace se pak budou u své rodičovské třídy registrovat pomocí třídní metody *registerInterface()* (viz str.23). Z tohoto návrhu vyplývá důležitá skutečnost, že tyto podtřídy se musí registrovat už při startu aplikace, jinak nebudou uvedeny v seznamu dostupných zařízení, který vrací metoda *getDevices()*. Java pro tyto případy poskytuje tzv. statický inicializační blok (obdoba konstruktoru třídy v C++), který se zavolá při prvním použití třídy a umožňuje inicializovat statické proměnné. Příklad použití statického inicializačního bloku:

```
public class ModularEEGInterface extends EEGInterface  {
   static {
       EEGInterface.registerInterface("ModularEEG", ModularEEGInterface.class);
   }
}
```

Pokud vyzkoušíme výše uvedený příklad, zjistíme, že třída ModularEEGInterface se nezaregistrovala. Příčinou je skutečnost, že virtuální stroj nahrává třídu ze souboru .class (a tím pádem i spouští statický inicializační blok) až při prvním volání statické (třídní) metody nebo konstruktoru této třídy. My ovšem obecně nemáme k dispozici žádný odkaz na tuto třídu a nechceme použít konfigurační soubor se seznamem dostupných rozhraní (a názvy tříd). To by znemožňovalo jednoduše doplňovat program o nové implementace rozhraní pouhým nakopírováním zkompilované třídy do adresáře uvedeného v cestě ke třídám (class-path) v manifestu souboru Jar s vyvíjenou aplikací.

Řešením je umístit všechny třídy rozhraní, které se mají nahrát do jednoho balíčku a explicitně zajistit jejich nahrání. K tomuto účelu byla vytvořena třída *PackageLoader,* pomocí které se všechny třídy v balíčku nahrají voláním statické metody loadPackage() s názvem příslušného balíčku. Příklad nahrání rozhraní :

```
PackageLoader.loadPackage("cz.vutbr.fit.Jfeedback.signal.interfaces");
```

Samotná implementace třídy *PackageLoader* nebyla triviální, protože Java neposkytuje žádnou přímočarou možnost zjistit, jaké třídy se v určitém balíčku nacházejí. Jedinou možností tak zůstává prohledávání souborového systému a zjišťování přítomnost souborů .class v podadresáři odpovídajícímu názvu balíčku. Tato skutečnost je dále komplikována variantou, kdy je aplikace spuštěna ze souboru .jar (a třídy rozraní jsou obsaženy uvnitř), nebo jsou třídy rozraní přidány do adresáře dostupného přes class–path (princip rozšiřujícího modulu, tzv. pluginu). Protože v současnosti je celý program kromě externích knihoven obsažen v jediném archivu .jar, implementoval jsem pouze první dvě možnosti a vyhledávání rozhraní v externích souborech .class a .jar ponechám na pozdější dobu. Pro obsáhlost zde neuvádím výpis zdrojového kódu třídy *PackageLoader*, čtenář jej může nalézt v souboru /Jfeedback/sources.zip na přiloženém CD.

Autodetekce zařízení a čtení dat z ModularEEG

Zařízení ModularEEG je připojeno k počítači přes sériové rozhraní a standardně komunikuje pouze jedním směrem zasíláním paketů speciálního protokolu do počítače. Pro implementaci jsem zvolil novější verzi 3, jejíž formát je následující:

```
// ModularEEG - popis formatu paketu verze 3
//
// 0ppppppx  hlavicka
// 0xxxxxxx
//
// 0aaaaaaa  kanal 0 LSB
// 0bbbbbbb  kanal 1 LSB
// 0aaa-bbb  kanal 0 a 1 MSB
//
// 0ccccccc  kanal 2 LSB
// 0ddddddd  kanal 3 LSB
// 0ccc-ddd  kanal 2 a 3 MSB
//
// 0eeeeeee  kanal 4 LSB
// 0fffffff  kanal 5 LSB
// 1eee-fff  kanal 4 a 5 MSB
//
// Popis:
```

```
//
// 1 a 0 = synchronizacni bity, Byte s 1 je vzdy v paketu posledni
// p = 6 bitove pocitadlo paketu
// x = bajt pomocnych kanalu
// a - f = 10 bitove vzorky z kanalu 0 - 5
// - = nevyuzito, musi byt nulove
```

Protože Java neobsahuje knihovny pro přístup k sériovému portu, použil jsem externí knihovnu RXTX (www.rxtx.org) s podporou několika platforem. Pokud chceme detekovat, ke kterému sériovému portu je zařízení připojeno, musíme postupně projít všechny porty v systému a pokusit se zachytit jedničkový synchronizační bit, který je na nejvyšší pozici v posledním Bytu paketu (viz protokol). Samotná detekce spočívá v načtení pole Bytů o velikosti jednoho paketu a zjišťování počtu těchto synchronizačních bitů v uvedeném poli. Po pěti neúspěšných pokusech přejdeme k dalšímu portu. Výsledek vyhledávání rozhlásíme všem příjemcům zprávy *EEGConnectionEvent*.

Filtrování signálu IIR a FIR filtry

Ke zjišťování úrovně signálu v jednotlivých pásmech potřebujeme potlačit kmitočty, které do těchto pásem nepatří. K tomu účelu jsem použil pásmové propusti složené s IIR a FIR filtrů, jejichž koeficienty jsem generoval pomocí programu Matlab. Příklad skriptu pro generování Butterwothových horních propustí 5. řádu s koeficienty typu double:

```
fid = fopen('high_pass.bin','wb');
for k = 1:50
  [b,a] = butter(5, k/128, 'high');
  fwrite(fid, a, 'real*8');
  fwrite(fid, b, 'real*8');
end
fclose(fid);
```

Při čtení výsledného binárního souboru však narazíme na problém s rozdílností pořadí ukládání Bytů do paměti u různých počítačových architektur. Existují normy little–endian (jako první se uloží nejméně významný Byte) a big–endian (opak, na nejnižší adrese je Byte s nejvyšším významem). V tomto případě zjistíme, že procesory Intel x86 a kompatibilní se řídí normou little–endian, kdežto virtuální stroj Javy pracuje s normou big–endian. Přímé čtení koeficientů ze souboru tedy není možné.

Nejelegantnějším řešením tohoto problému je použití třídy *ByteBuffer* z balíku java.nio (New Input–Output), která umí metodou *wrap()* zapouzdřit pole Bytů a číst z něj hodnoty primitivních datových typů podle nastaveného pořadí Bytů metodou *order()*. Příklad čtení koeficientu typu double z pole *aux_buf* upořádaného podle normy little–endian:

```
koef = ByteBuffer.wrap(aux_buf).order(ByteOrder.LITTLE_ENDIAN).getDouble();
```

Na rozdíl od vždy stabilních filtrů FIR je funkčnost filtrů typu IIR ohrožena jejich nestabilitou. Ta je způsobena rozkmitáním zpětných vazeb, které filtr obsahuje. V mém případě se podařilo implementovat stabilní filtry IIR pouze druhého řádu. Se zvyšujícím se řádem filtru a klesajícím kmitočtem stabilita filtrů klesala. K vůli tomu jsem ve třídě *RMSBandSignalBlock* (viz str. 23) použil pouze filtry typu FIR.

7.2 Implementace grafické části

Aktivní renderování

Aktivní renderování [19] je technika pro překreslování scény, kdy se nepoužívá standardní vykreslování poskytované knihovnou grafického uživatelského rozhraní oknu se scénou, ale v separátním vlákně se spouští následující cyklus:

1. Aktualizujeme zobrazované elementy
2. Vykreslíme scénu na obrazovku
3. Uspíme na krátkou dobu vykreslovací vlákno
4. Přejdeme zpět k bodu 1

Uvedený algoritmus jsem použil ve třídě *GameWindow*. Výhodou tohoto postupu je skutečnost, přímo řídíme cyklus vykreslování, nemusíme někdy zbytečně čekat na vlákno uživatelského rozhraní a hlavně tento postup je velmi výhodný pro vykreslování sprajtů a animací.

Přepínání rozlišení grafického výstupu

U třídy *GameWindow* se vyskytl problém s přepínáním rozlišení a barevné hloubky celoobrazovkového okna pomocí třídy *GraphicsDevice*. Scéna se do zadaného rozlišení přepnula, ale zobrazené barvy neodpovídaly skutečnosti a navíc se na obrazovce objevily body a proužky, které tam nepatřily. Tento problém vyřešilo přidání zpoždění vlákna o 40 ms, než byla zapnula podpora dvojitého stránkování grafické paměti (double buffering). Způsob přepínání rozlišení ilustruje následující kód:

```
// obdrzime referenci na graficke zarizeni
_device =
       GraphicsEnvironment.getLocalGraphicsEnvironment().getDefaultScreenDevice();
_device.setFullScreenWindow(this);

// pokud je podporovana zmena grafickeho rezimu
if (_device.isDisplayChangeSupported()) {
       try {
              _device.setDisplayMode(dm);
       }
       catch (IllegalArgumentException ex) {
              System.out.println("Chyba grafickeho rezimu!");
              System.exit(1);
       }
}

// cekani na prepnuti barevne hloubky
try {
       Thread.sleep(40);
}
catch (InterruptedException e) {};

// zapnuti doublebufferingu
setSize(dm.getWidth(), dm.getHeight());
createBufferStrategy(2);
```

Rychlost vykreslování scény

Při implementaci jsem měl příležitost vyzkoušel některé pokročilé grafické funkce třídy *Graphics2D,* které jsou zde nově obsaženy oproti starší třídě *Graphics*. Konkrétně se jednalo o barevný přechod, který realizuje třída *GradientPaint*, průsvitné kreslení s třídou *AlphaComposite* a vyhlazování textu aktivované přímo u třídy *Graphics2D*. Pokud bych měl hodnotit rychlost vykreslování jednotlivých funkcí, tak vyhlazování textu ani průsvitné kreslení výkon příliš neovlivnilo. Naproti tomu barevné přechody zpomalovaly vykreslování natolik, že se scéna nestačila plynule překreslovat a bylo je nutné z programu zcela vypustit. Celkově není rychlost vykreslování v Javě příliš vysoká a bylo by vhodné použít nějakou multiplatformní knihovnu s nativní částí, která má přímý přístup ke grafické kartě. Jako nejzajímavější se jeví použití knihoven implementujících rozhraní OpenGL, které nám kromě zrychlení vykreslování umožní přejít do trojrozměrného zobrazení.

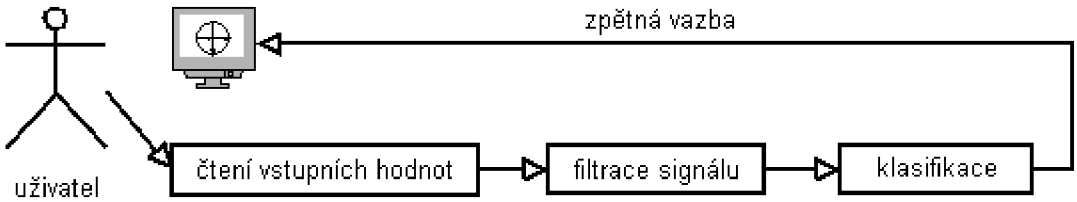
8. Návrh ovládání směru pomocí EEG zařízení

Otázka ovládání funkcí počítače pomocí lidského myšlení se stala v posledních letech jedním z velmi populárních témat u odborné i laické veřejnosti. Známy jsou systémy umožňující procházet se ulicí města utvořeného virtuální realitou nebo prostřednictvím počítače komunikovat postiženým lidem se svým okolím. Tato oblast výzkumu však i přes výrazné pokroky stojí stále před mnoha úskalími a problémy, které čekají na rozřešení. Doposud se nepodařilo nalézt a sestavit zcela univerzální, běžně dostupný a hlavně zcela přesný systém zvládající všechny úlohy spojené se stykem člověka a stroje.

8.1 Teorie rozhraní BCI

Činnost typického rozhraní BCI (Brain-Computer Interface, rozhraní lidského mozku a počítače) [20] se dá rozdělit do čtyř částí:

- získávání vstupních dat – bioelektrická aktivita mozku je zaznamenána EEG zařízením a zaslána do počítače
- předzpracování signálů – potlačují se nežádoucí artefakty a upravuje průběh signálů pro co nejsnadnější získání požadované informace
- klasifikace – vlastní získávání informace pomocí neuronových sítí, skrytých Markovových modelů, Bayesiánských sítí a dalších prostředků umělé inteligence
- zpětná vazba – prezentace výsledku klasifikátoru prostřednictvím počítače, tímto dochází ke zpětnému ovlivnění uživatele systému

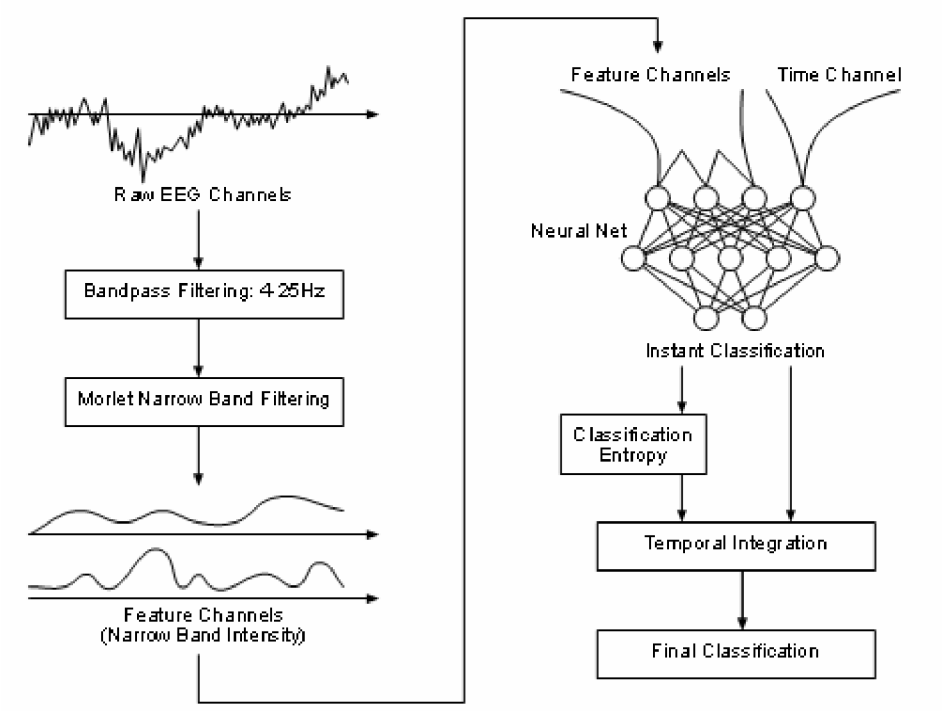


Obr.8.1. Schéma rozraní lidský mozek – počítač

Nejzákladnější rozdíl mezi systémy panuje v přístupech při získávání vstupních dat. Existují dva způsoby umístění elektrod:

- invazivní – člověku jsou voperovány elektrody přímo do hlavy na povrch mozkové kůry. Vyloučí se tím zkreslení a zeslabení signálů při průchodu lebkou a pokožkou hlavy. Tento způsob je vhodný, pokud je rozhraní používáno pořád (např. u postižených lidí) nebo v případě nutnosti mít silný vstupní signál.
- neinvazivní – elektrody jsou umístěny na povrchu hlavy, operace tudíž není nutná. Tento způsob je nejvhodnější pro příležitostné použití.

Jako konkrétní příklad neinvazivního systému obsahujícího inteligentní klasifikátor může být uveden program ABI BCI od Pedra Ortegy [21] (viz obr.8.2.) Vstupní část obsahuje pásmovou propust 4-25Hz, na kterou navazuje blok s vlkovou transformací pomocí Morletovy vlnky. Výsledkem je skupina pásem, která spolu s časovým signálem vstupuje do dvouvrstvé dopředné neuronové sítě. Výstup sítě je váženě sumován podle času, kdy váha každého prvku závisí na entropii informace. Výsledná klasifikace neuronové sítě s vyšší entropií ukazuje na značnou nejistotu výsledku a je tudíž sumovány s nízkou váhou. Podobně výsledkům s nízkou hodnotou entropie je přiřazena vyšší váha, protože je zde větší pravděpodobnost správného určení vzoru.

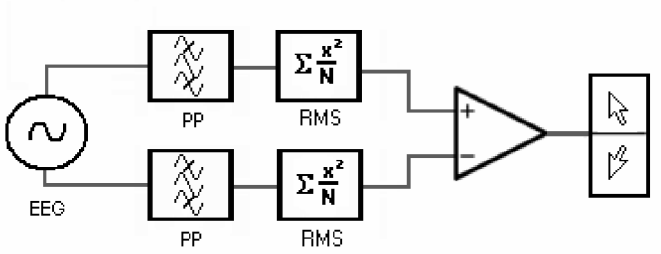


Obr.8.2. Příklad komplexního systému BCI [21]

8.2 Konkrétní návrh ovládání počítače

Pro svůj návrh ovládání kurzoru jsem vyšel z koncepce zcela odlišné od příkladu uvedeného výše, protože omezený počet kanálů zařízení ModularEEG by neumožňoval dostatečně efektivní využití neuronových sítí či jiných prostředků umělé inteligence. Důležitou inspirací pro mne byl poznatek z oblasti biologické zpětné vazby a psychostimulátorů o tom, že člověk je schopen vůlí cíleně ovlivňovat úroveň bioelektrické aktivity mozku v určitém pevně stanoveném kmitočtovém pásmu, pokud je mu poskytnuta informace o rychlosti a průběhu změny kmitočtu snímaného signálu.

Základní princip metody je poměrně jednoduchý (viz obr. 8.3.). Spočívá v průběžném porovnávání střední hodnoty výkonu signálu (RMS) dvou pásem získaných filtrací přes pásmovou propust (PP). Výsledek porovnání ovlivní směr pohybu kurzoru pro danou osu souřadného systému – tedy např. nahoru nebo dolů. Kmitočty pásem jsou záměrně voleny velmi blízko sebe a filtry mají nastavenu úzkou šířku propouštěného pásma, aby uživatel mohl snadno ovlivňovat pohyb kurzoru pomocí „přelaďování“ mozkových vln .

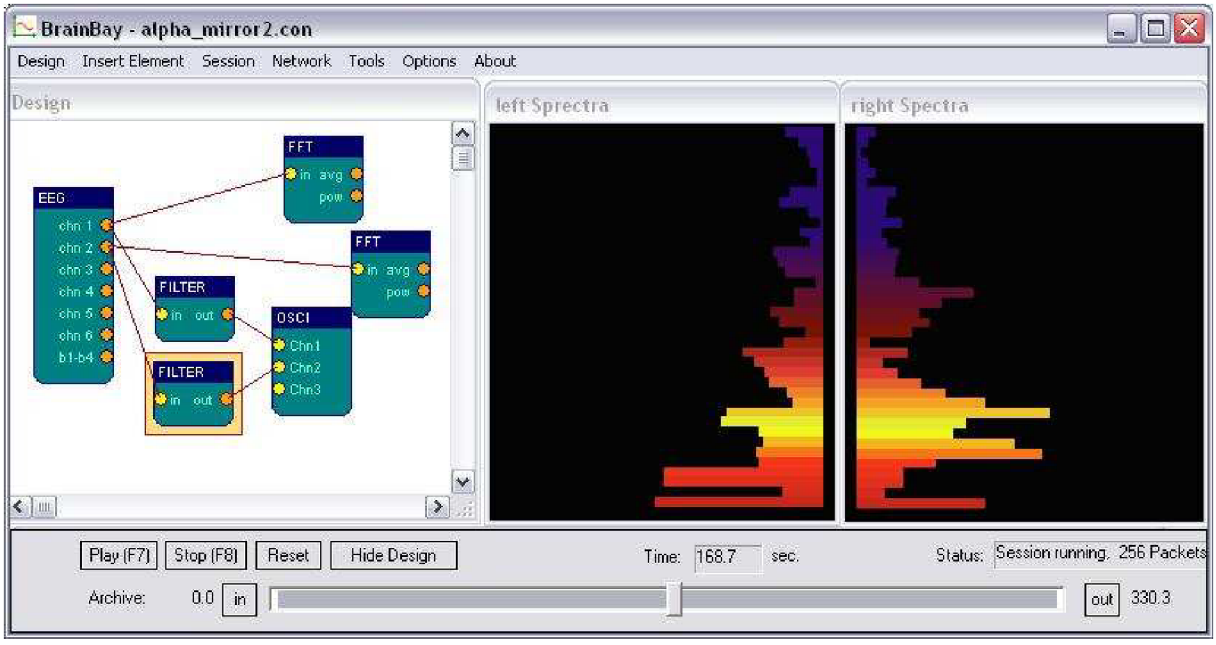


Obr.8.3. Principiální schéma ovládání pohybu kurzoru

K implementaci rozhraní mi posloužil program BrainBay [22] (viz obr.8.4.), který je vyvíjen v rámci projektu OpenEEG skupinou autorů kolem Christopha Veigla. Tato aplikace má širokou škálu možností a je velmi vhodná pro experimentování se zpracováním bioelektrických a obrazových signálů. Pro získání představy o schopnostech tohoto programu zde uvedu alespoň souhrn jeho základních funkcí:

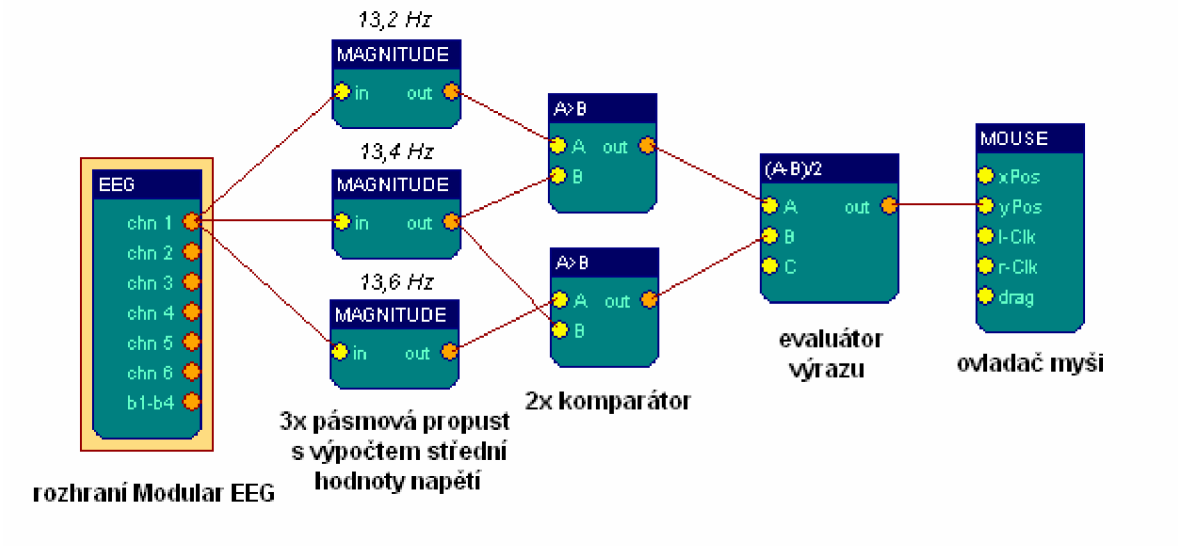
- digitální filtrování signálu, výpočet FFT se zobrazením spektrogramu
- funkční boky s mat. funkcemi (korelace, komparátor, integrátor, výpočet výrazů a další)
- multimediální výstup (přehrávání souborů midi/wav/avi, zobrazování grafů, ...)
- posílání dat počítačovou sítí (používá se Neuroserver framework)
- čtení a ukládání souborů ve formátu EDF (European Data Format)
- podpora webkamer a detekce pohybů tváře (pro vládání kurzoru pomocí hlavy)
- funkce k ovládání počítače (ovládání kurzoru myši, generování stisku kláves)

Pro náš cíl je obzvláště užitečná funkce ovládání kurzoru myši, s jejíž pomocí lze v programu vytvořit fungující rozhraní pracující pouze s filtry signálu a komparátory.

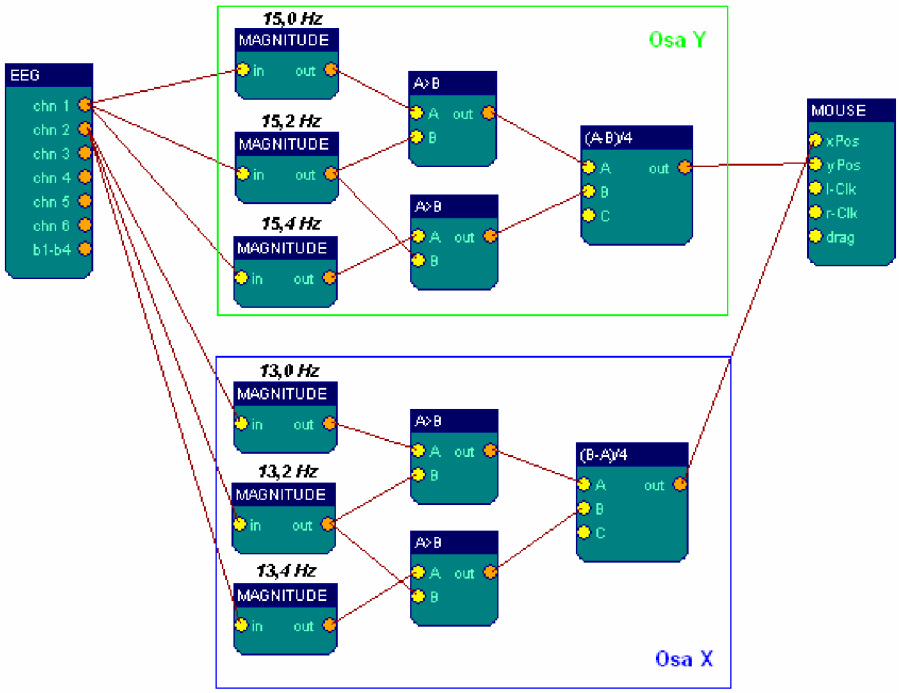


Obr.8.4. Ukázka obrazovky programu BrainBay [22]

Vlastní implementované schéma v programu BrainBay lze vidět na obrázku 8.5. Oproti původnímu návrhu bylo zapojení vylepšeno o třetí pásmo (uprostřed), které je neutrální na pohyb kurzoru a umožňuje kurzor zastavit (jinak by kurzor stále pulsoval nahoru a dolů). Ze vstupního rozhraní je signál veden do 3 bloků zjišťujících úroveň napětí v daných pásmech (střední kmitočet je zobrazen vždy nad blokem, šířka pásma je nastavena na 0,2 Hz). Hodnoty úrovně napětí jsou připojeny na vstup komparátorů, které v případě splnění zobrazené podmínky propustí na výstup hodnotu vstupu A, v opačném případě je výstup odpojen. Výstupy obou komparátorů jsou odpojeny v okamžiku, kdy je nejvyšší úroveň signálu v prostředním pásmu a chceme zastavit pohyb kurzoru. Evaluátor výrazu obě hodnoty z komparátorů odečte a polarita výsledku je použita k ovládání směru kurzoru. Číslo ve jmenovateli výrazu slouží ke zpomalení rychlosti jeho pohybu.



Obr.8.5. Vylepšené zapojení pro ovládání ypsilonové souřadnice myši pomocí ModularEEG v programu Brainbay



Obr.8.6. Schéma ovládání směru rozšířeného na obě osy obrazovky

Úspěch s ovládáním kurzoru v jednom směru mne inspiroval k duplikaci použitých bloků a rozšíření schématu o ovládání směru v ose x. Vstupní signál je odebírán z druhého kanálu zařízení ModularEEG, který je připojen na opačnou mozkovou hemisféru, než první kanál. Aby se navzájem obě osy neovlivňovaly, jsou pro osu x zvoleny jiné kmitočty pásem. Výsledné schéma zachycuje obrázek 8.6.

8.3 Výsledky experimentů

Získané zkušenosti z provedených experimentů s ovládáním počítače pomocí EEG zařízení lze shrnout do následujících bodů:

- Je nutné, aby se uživatel při ovládání dobře soustředil. Únava má na výsledek značně negativní vliv.

- Zařízení ModularEEG nemá zabudováno měření přechodové impedance elektrod, je tedy před každým zahájením snímání nutné řádně zkontrolovat správnost připojení elektrod pomocí bloku osciloskop v programu BrainBay. Postačuje vizuální kontrola průběhu signálu, který by měl s malou amplitudu periodicky kmitat kolem nulového bodu.

- Kmitočty pásem bloků Magnitude je vhodné volit v rozsahu přibližně 7 až 18 Hz, mimo tento rozsah se obtížnost ovládání zvyšuje. Šířka sousedních pásem by se neměla překrývat.

- Pokud chybí prostřední „mrtvé“ pásmo, je velice těžké udržet kurzor na jednom místě, ten se neustále pohybuje po obrazovce.

- Ovládání obou směrů pohybu kurzoru je o poznání obtížnější, než v případě ovládání pouze jednoho směru. Snadnější je ovládání v tom směru, jehož kanál je připojen na dominantní hemisféru (u praváků je to levá hemisféra).

- Bylo by velmi vhodné zvýšit vzorkovací frekvenci zařízení ModularEEG, neboť je možno pozorovat určitou prodlevu v odezvě na povely uživatele.

9. Závěr

Jedním z cílů této práce se stala snaha vytvořit volně šiřitelný program pro EEG biofeedback určený široké veřejnosti. Důležitým faktorem je podpora zařízení ModularEEG, vyvíjeného v rámci projektu OpenEEG. V souboru software dostupného v tomto projektu dosud chybí taková aplikace, která by byla zaměřena čistě na EEG biofeedback a vytvořila spolu se zařízením ModularEEG jeden celek s dostupnými zdrojovými kódy a konstrukční dokumentací. Tento fakt mne přiměl k vývoji vlastní aplikace v jazyce Java.

Programování desktopových aplikací pro platformu jazyka Java je stále velmi málo rozšířeno a v současné době existuje pouze nemnoho programů napsaných čistě v tomto jazyce v porovnání s ostatními programovacími jazyky. Navržený program Jfeedback by mohl tuto množinu rozšířit a ukázat, že Java není vhodná jenom pro programování appletů a serverových aplikací. Distribuce zdarma formou otevřených zdrojových kódů dává možnost ostatním uživatelům tento program upravovat a dále volně rozšiřovat, což opět o malý kousek posílí pozici open−source software.

Z časových důvodu byl vytvořen pouze základní funkční prototyp programu Jfeedback. Během testování se jako hlavní problém ukázala především pomalost vestavěné knihovny pro 2D grafiku při zapnutém vyhlazování zobrazovaných textů a kreslení průhledných obrázků. Nejlépe by se tato nepříjemná skutečnost dala obejít přechodem na zobrazování ve 3D a použitím některé z rozšiřujících knihoven pro práci s grafickým rozhraním OpenGL (JOGL, LWJGL…). Vyřešila by se tak jednak otázka výkonnosti vykreslování, dále by se pak výrazně zvýšila uživatelská atraktivita programu. Mezi důležité kroky v dalším rozvoji aplikace patří také doplnění grafického výstupu o výstup zvukový, což by mohlo napomoci většímu vtažení uživatele do hry a zlepšení efektivity EEG biofeedbacku. Uživatelskému rozhraní by také prospělo vylepšení knihovnami platformy NetBeans nebo freewareovou knihovnu JGoodies Looks, aby se vzhled vyvíjené aplikace přiblížil komerčním produktům. Pro vzbuzení zájmu u zahraničních uživatelů je velmi důležité připravit anglickou verzi a případné lokalizace do dalších světových jazyků. Časem by se mohla přidat podpora i pro jiná zařízení, než je ModularEEG.

Dalším z témat, kterým jsem se ve této práci zabýval, byla možnost ovládání směru kurzoru myši pomocí lidského mozku. Motivací k řešení tohoto problému byla snaha nahradit klasická vstupní periferní zařízení ovládacím přístrojem vhodným i pro postižené lidi, kteří jinak nejsou schopni používat počítač. Výsledkem této práce by měl být návrh principu zařízení, které čte signál z elektroencefalografu a na základě určitého algoritmu rozhodne, kterým směrem se má kurzor pohybovat.

Pro své experimenty s ovládáním směru kurzoru jsem opět použil zařízení ModularEEG. Tento přístroj má oproti profesionálním encefalografům pouze malý počet kanálu, a tudíž je požití klasických prostředků umělé inteligence, jako jsou například neuronové sítě, pro rozpoznávání zvoleného směru pohybu velmi komplikované. Můj návrh byl inspirován tzv. psychowalkmany, které se snaží audiovizuálně stimulovat určitá pásma mozkových vln podobným způsobem jako EEG biofeedback, ale s tím rozdílem, že zde není použita elektroencefalografie a biologická zpětná vazba. Základním bodem návrhu je poznatek, že člověk je v malém rozsahu schopen rychle přelaďovat kmitočet svých mozkových vln. Pomocí této skutečnosti jsem navrhnul princip ovládání, který spočívá

v porovnávání úrovně signálu ve dvou úzkých kmitočtových pásmech vzdálených nepříliš daleko od sebe.

Výsledky experimentů s tímto rozhraním lze v současnosti považovat za slibné, avšak k dokonalému ovládání počítače vede ještě dlouhá cesta. Při testování se vyskytnul problém s kmitáním kurzoru nahoru a dolů bez možnosti jeho stabilizace, který byl vyřešen přidáním třetího „neaktivního“ pásma mezi dvě stávající. Tedy pokud je nejvyšší úroveň signálu v tomto pásmu, tak kurzor stojí na místě. Dále byla pozorována prodleva v odezvě na uživatelovy povely, která je pravděpodobně způsobena zpracováním signálu v počítači a mohla by být odstraněna zvýšením vzorkovací frekvence v EEG zařízení.

Další rozvoj výsledků tohoto návrhu by mohl přinést intenzivnější výzkum. Zajímavým nápadem by mohla být konstrukce malého bezdrátového zařízení umístěného na hlavě, s jehož pomocí by se dal ovládat počítač. Přichází doba snah o určité „zlidšťování“ počítačů, které jsou v současné době stále ještě chápány jako neživé „hromádky“ elektronických součástek, pomocí počítačového vidění a hlasové komunikace. Začleněním těchto komunikačních kanálů by mohlo vzniknou komplexnější víceúčelové rozhraní člověk–stroj. Ještě bude nějakou dobu však bude trvat, než odložíme myš s klávesnicí a vstoupíme do nové éry ve vývoji počítačů.

Použité materiály:

[1] Kněžík, J.: *EEG biofeedback rozhraní lidského mozku a počítače* [Semestrální projekt]. FIT, VUT v Brně, 2006

[2] Zentiva, a.s, Praha: *Proč k epilepsii dochází a také něco k lidskému mozku*. Článek dostupný na URL https://www.zdravcentra.cz/cps/rde/xchg/zc/xsl/8514_3711.html (leden 2006)

[3] *Převrat v pohledu na mozek!*. časopis 21. století, 1, 2006, s. 68–75

[4] Churý, L.: *UMĚLÁ INTELIGENCE - 2*. Článek dostupný na URL http://programujte.com/view.php?cisloclanku=2005061201 (leden 2006)

[5] Holčík, J., Straszecka, E.: *Bionika (Biologické systémy a procesy)*. UBMI, FEI, VUT v Brně, 1999, s. 37–51

[6] Kučera, F.: *Realizace analogové části EEG rozhraní* [Diplomová práce]. FEI, VUT v Brně, 2000

[7] *Electroencephalography - Wikipedia, the free encyclopedia*. Článek dostupný na URL http://en.wikipedia.org/wiki/Electroencephalography (prosinec 2005)

[8] Louis, S.: *EEG waves as defined by frequency*. Článek dostupný na URL http://www.brown.edu/Departments/Clinical_Neurosciences/louis/eegfreq.html (prosinec 2005)

[9] *Biofeedback - Wikipedia, the free encyclopedia*. Článek dostupný na URL http://en.wikipedia.org/wiki/Biofeedback (prosinec 2006)

[10] L. Churý: *Biofeedback*. Článek dostupný na URL http://programujte.com/view.php?cisloclanku=2005120301 (leden 2006)

[11] Tyl, J., Tylová, V.: *LEHKÉ MOZKOVÉ DYSFUNKCE: Nové metody nápravy*. Biofeedback institut, Praha 2003. Dokument dostupný na URL http://www.eegbiofeedback.cz/cesky/cesky.php?menu=stazeni (prosinec 2005)

[12] *Psychowalkman, AVS – Audiovizuální stimulační přístroj, Galaxy Czech, s.r.o. Staženo* z URL www.psychowalkman.cz (prosinec 2006)

[13] *EEG Biofeedback*. Staženo z URL http://www.eegbiofeedback.cz/cesky/cesky.php (listopad 2005)

[14] The Learning Curve, Inc., PA, USA: *Brain patterns*. Článek dostupný na URL http://www.brain-trainer.com/brain_patterns/ (prosinec 2006)

[15] Biofeedback intitut, Praha: *Brainfeedback III, Uživatelská příručka programu*. Dokument dostupný na URL http://www.eegbiofeedback.cz/cesky/cesky.php?menu=stazeni (prosinec 2005)

[16] Gamma, E., aj.: *Návrh programů pomocí vzorů*. Grada Publishing, 2003

[17] Schmuller, J.: *Myslíme v jazyku UML.* Grada Publishing, 2001.

[18] Jan, J.: *Číslicová filtrace, analýza a restaurace signálu.* Vutium, 2003

[19] Brackeen, D.: *Vývoj her v jazyku Java.* Grada Publishing, 2004

[20] Gibbs, M.: *Research page.* Článek dostupný na URL http://www.robots.ox.ac.uk/~mgibbs/research.html (prosinec 2006)

[21] Ortega, P.: *Experimental Brain Computer Interface Software for the ModularEEG.* Článek dostupný na URL http://www.dcc.uchile.cl/~peortega/abi/ (prosinec 2006)

[22] Veigl, Ch: *BrainBay – User Manual*. Dokument dostupný na URL http://www.shifz.org/brainbay/ (leden 2007)

Obsah přiloženého CD

Na přiloženém kompaktním disku čtenář nalezne text této technické zprávy, uložený ve formátu Microsoft Word a Adobe Acrobat. V adresáři *Jfeedback* se nacházejí zdrojové kódy (soubor *sources.zip*) a instalátor (*setup.exe*) programu Jfeedback, vyvíjeného pro podporu open–source aplikací pro EEG biofedbacku.